DAHLGREN DIVISION
NAVAL SURFACE WARFARE CENTER
Silver Spring, Maryland 20903-5640

NSWCDD/TR-95/125

MISSION CRITICAL SYSTEM DEVELOPMENT: DESIGN VIEWS AND THEIR INTEGRATION-- VERSION 2.0

BY NGOCDUNG HOANG   NICHOLAS KARANGELEN   STEVEN HOWELL

SYSTEMS RESEARCH AND TECHNOLOGY DEPARTMENT

JUNE 1996

19960627 000

DTIC QUALITY INSPECTED 1

REPORT DOCUMENTATION PAGE

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, search existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services. Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 1996	Final

4. TITLE AND SUBTITLE

Mission Critical System Development: Design Views and Their Integration--Version 2.0

5. FUNDING NUMBERS

6. AUTHOR(s)

Ngocdung Hoang, Nicholas Karangelen, and Steven Howell

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Commander
Naval Surface Warfare Center, Dahlgren Division
White Oak Detachment (Code B40)
10901 New Hampshire Avenue
Silver Spring, MD 20903-5640

8. PERFORMING ORGANIZATION REPORT NUMBER

NSWCDD/TR-95/125

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Naval Research (Code 4411B)
800 North Quincy Street
Arlington, VA 22217-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

This document describes a formalism that allows for the capture and analysis of very large, complex, computer-based, real-time systems. The formalism covers all aspects of a system, including functional (the functions a system performs) and non-functional (characteristics of system performance) attributes. The current proposed formalism captures the system design in five different capture views: Informational, Functional, Behavioral, Implementation, and Environmental. Each capture view explores different aspects of the system, and all five in total provide a more complete understanding of the system design. Although the ultimate goal is to capture all aspects of the system in the five views, it is understood that techniques for analyzing system attributes such as hard real-time, security, reliability, and dependability are very diverse and evolve rapidly. Hence, the captured information is expected to be extendible and customizable to fulfill the need of each individual project. An example of a passive submarine sonar system is used throughout the document to illustrate the various capture techniques.

14. SUBJECT TERMS

system capture; system analysis; very large, complex, computer-based real-time systems; formalism; informational, functional, behavioral, implementation, and environmental capture views

15. NUMBER OF PAGES

247

16. PRICE CODE

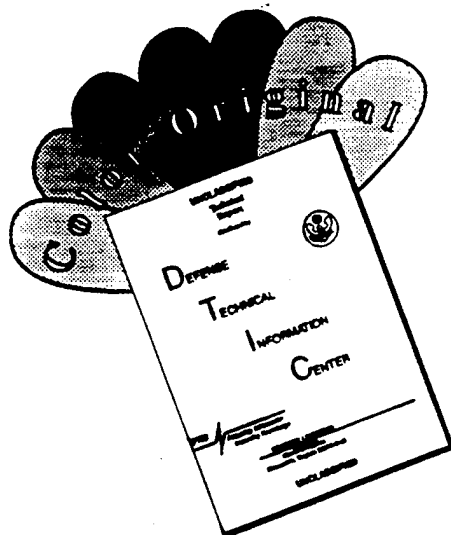| 17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500   i/ii   Standard Form 298 (Rev 2-89)
Prescribed by ANSI std. Z39-18
298-102

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF COLOR PAGES WHICH DO NOT REPRODUCE LEGIBLY ON BLACK AND WHITE MICROFICHE.
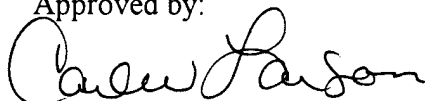
FOREWORD

This document describes a formalism that allows for the capture and analysis of very large, complex, computer-based, real-time systems. The formalism covers all aspects of a system, including functional (the functions a system performs) and non-functional (characteristics of system performance) attributes. Non-functional attributes of a system, such as timing, dependability, security, and reliability, should be captured and analyzed at the early stage of the system development process to guarantee the correctness of a system's performance. The current proposed formalism captures system design in five different views such that analysis can be correctly performed. The capture views are Informational, Functional, Behavioral, Implementation, and Environmental. Each capture view explores different aspects of a system, and all five in total provide a more complete understanding of the system. Although the ultimate goal is to capture all aspects of a system in the five views, it is understood that techniques for analyzing system attributes such as hard real-time, security, reliability, and dependability are very diverse and evolve rapidly. Hence, the captured information is expected to be extendible and customizable to fulfill the need of each individual project. This document updates Mission Critical System Development: Design Views and their Integration (NAVSWC TR 91-586).

This research was performed under the Real-Time System Design Capture and Analysis thrust within the Systems Design Synthesis Technology (SDST) task. The task is part of the Engineering of Complex Systems (ECS) technology project at Naval Surface Warfare Center Dahlgren Division (NSWCDD), which is sponsored by the Office of Naval Research (ONR).

The authors would like to thank their sponsor, ONR Code 311, and Ms. Elizabeth Wald, in particular. In addition, the authors would like to thank Mr. Cuong Nguyen and the Anti-Submarine Warfare (ASW) Methodology and Tools group, under the direction of Mr. Janis Bilmanis, for their technical support.

Approved by:

CARL W. LARSON, B40, Head
Advanced Systems Research and Technology Group

CONTENTS

CONTENTS (Cont.)

ILLUSTRATIONS

TABLES

CHAPTER 1

INTRODUCTION

This document describes a formalism that allows for the capture and analysis of very large, complex, computer-based, real-time systems. The formalism covers all aspects of the system including functional (the functions a system performs) and non-functional (characteristics of system performance) attributes. Non-functional attributes of the systems, such as timing, dependability, security, and reliability, should be captured and analyzed at the early stage of the system development process to guarantee the correctness of a system's performance. The current proposed formalism captures the system design in five different views such that analysis can be correctly performed. The capture views are Informational, Functional, Behavioral, Implementation, and Environmental. Each capture view explores different aspects of the system and all five in total provide a more complete understanding of the system. Although the ultimate goal is to capture all aspects of the system in the five views, it is understood that techniques for analyzing system attributes such as hard real-time, security, reliability, and dependability are very diverse and evolve rapidly. Hence, the captured information is expected to be extendible and customizable to fulfill the need of each individual project.

1.1 PROBLEM

The requirements for large-scale, mission critical systems have increased drastically in response to technology gains and advanced threats. These requirements, which include real-time, time-critical, fault tolerant and security issues, have put large burdens on the systems engineers in designing these highly complex systems. Even though significant increases have been made in the capacity and capability of hardware, software, and human resources (i.e., human computer interface), the engineering capability to engineer systems which effectively embody these components has not kept pace, resulting in a failure to fully utilize their potential capabilities.

A variety of methodologies has been described for characterizing and capturing complex system designs. These methodologies have been developed by systems engineers and academicians from both theoretical and practical viewpoints. Several different perspectives of the systems engineering design and capture problem exist. One popular perspective is the Yourdon-DeMarco style Structured Analysis methodology which defines a hierarchy of functions and data flows to describe the system.[1] Real-time extensions, such as those developed by Hatley and Pirbhai[2] and Ward and Mellor,[3] integrate control flow with data flow. An object-oriented approach is embodied in the Information Model approach described by Shlaer and Mellor.[4] Hardware and software engineers often perceive the system through the various hardware development formalisms (e.g., VHDL[5]) and software development methods and tools[6] which

address the design and analysis of specific implementations. Personnel with operational experience using earlier generations of the system under design (such as naval officers in the case of an antisubmarine warfare (ASW) combat system) tend to see the system from an operator's point of view and within the context of one or more operational scenarios.

Unfortunately, current methodologies do not fully support large, time-critical, real-time systems. For example, they fail to adequately address critical information that must be specified in order to understand and analyze the system under design. Particularly, non-functional attributes and estimation parameters for analysis are not formally specified in current methods. Part of this stems from the fact that many of the methods were originally constructed for software development for systems developed in uniprocessor environments. Large, complex, real-time systems are typically developed on parallel and distributed hardware, which are a combination of computers, sensors, and weapons. These methods also tend to de-emphasize non-functional (performance-oriented) hardware and the distribution aspects that make it difficult to analyze the systems.

In addition, scalability of the methods and their automation is an important issue for large, complex systems. Methods that are applicable for small and moderate-sized systems become inadequate for large systems which are typically developed by hundreds of systems developers and implemented in millions of lines of source code. Reasons for the failures of the methods include the structure and mechanization of the methods and the complexity of the algorithms within the methods.

Additionally, most of the current methods are being developed independently. While each method explores certain aspects of the system, each has its own format, notation, and conceptual basis. Transition techniques, which allow the information to be retained when moving from one method to another, are not available. This limitation forces systems engineers to completely recapture the system information for every evaluation technique. The recapture effort is time consuming, costly, and frequently generates inconsistent information between different captured models; therefore, system engineers could end up with incomparable analysis results and would not be able to comprehend the total effect of their design.

1.2 BACKGROUND

This research was performed under the Real-Time System Design Capture and Analysis thrust within the Systems Design Synthesis Technology (SDST) task. The task is part of the Engineering of Complex Systems (ECS) technology project. The work is sponsored by the Office of Naval Research (Code ONR-311). The overall objective of the ECS project is to provide an integrated, system level, full life cycle engineering methodology, supporting mechanization and engineering environments for the next generation of large-sized and complex systems. Within this objective, the SDST task is attempting to develop and enhance the system level ability to specify, capture, synthesize, analyze, and prototype the design of complex, real-time, time-critical, fault tolerant and secure systems.

This research is tightly associated with other ongoing efforts within the ECS project. Inputs to this research are structured requirement documents (the format of which can be either English text or some formal graphical representation) generated by the Requirement Specification and Traceability thrust. Captured information will be available to other researchers within the System Design Synthesis Technology task, as well as other tasks in the project (Modeling, Re-engineering).

Currently the closest coordination is with the Design Structuring and Optimization Allocation thrust within the SDST task. The selected design information is captured and transitioned into the optimization process. The transition techniques are expected to support other evaluation techniques as well. Hence, captured information could be effectively used by the Design Synthesis for High Assurance and the Systems Modeling thrust. This thrust also tightly relates to the System Requirement and Traceability thrust. Once the system requirements are organized in a clear structured format and are well analyzed by the Requirement Specification and Traceability thrust, they are used by the Capture thrust as input to generate the design. Figure 1-1 shows the different tasks within the ECS project and Figure 1-2 shows the relationship between thrusts within the SDST task.

The ongoing research is exploring ways to allow systems engineers to formally specify a system's design given a set of requirements. Capabilities to examine and analyze the consistency, completeness and correctness of the design is being matured. Techniques for providing the necessary detailed information and allowing the system design to move through the design process (whether it is the Waterfall, Modified Waterfall, Spiral, Rapid Prototyping, etc.) are being developed. The effort hopes to also allow systems engineers to optimize the design's distribution, decomposition, and allocation from requirements before deciding which parts will be allocated to hardware, software or performed by humans (humanware).

1.3 DESCRIPTION OF EXAMPLE

An example of a passive submarine sonar system will be used throughout this document to illustrate the various capture techniques. This example was developed by Molini[7] to provide a meaningful test problem for research of real-time distributed systems. The following paragraphs briefly describe the passive sonar system example.

Sonar equipment is used for determining the presence, location, or nature of objects in the sea from the sound that the objects emit or reflect. Active sonar transmits an acoustic signal which, when reflected from a target, provides the sonar receiver with a signal. Based on this signal, detections and position estimates are made. Passive sonar on the other hand bases its detection and estimation on sounds that emanate from the target itself.

FIGURE 1-1. ENGINEERING OF COMPLEX SYSTEMS PROGRAM

FIGURE 1-2. SYSTEMS DESIGN SYNTHESIS PROJECT

In passive sonar, the received acoustic waveform from each hydrophone consists of one or more signals and background noise. The hydrophone converts the acoustic waveform to an electronic signal. The signals are amplified, filtered, sampled, and digitized in a signal conditioner. The digitized hydrophone outputs from the signal conditioner are combined by a digital beamformer to form a set of beams. (A beam emphasizes the sound from the beam direction and reduces the sound from other directions.) Beam data are then processed to obtain detection and estimation statistics. Based on the values of these statistics, the system decides where targets are located. Detected targets are tracked by modifying the beamformer parameters and steering a beam toward the target.

A detected target is also analyzed by classification. This includes distinguishing a returned signal from a target with regard to the type of target that produced the signal and by a signal frequency spectrum and dynamics (e.g., a school of fish versus a submarine) of its target signal.

The system is designed to detect and track submarine targets that can operate at any speed from 0 to 15 knots at any depth from 0 to 500 feet. The target sound source is assumed to consist of vibrational harmonics from rotating machinery with a nominal rotation rate of 2,400 rpms. Therefore, it is expected that the return signal from a submarine contains a fundamental 40-hertz component and the first three harmonics.

The data processing load for a sonar system is roughly proportional to the number of hydrophones, number of beams, and the sample rate (i.e., the larger the array of hydrophones, the greater the beamforming gain and the greater the detection range; the finer the beam width, the finer the display resolution that assists in resolving multiple targets even though they appear close together). Some of the beams are used for detection displays, some for tracking targets, and some just for listening. Typically, the number of detection beams is fixed, so the detection processing varies little over time. The number of steered tracker beams may change as targets come and go. When the number of tracker beams is changed, the tracking load changes proportionally.

CHAPTER 2

OVERVIEW OF DESIGN CAPTURE

In order to completely capture a design, a well-defined methodology should be provided to systems engineers. This methodology must be robust enough to support the capture of the general design information as well as the detailed information that is required for various evaluation methods to be performed early on in the design process. In other words, both the quantity and the quality of the design elements must be captured appropriately and maintained throughout the development process. This capability will support systems engineers to manage requirement changes and/or new technologies insertion during the design and to guarantee the consistency of the analysis results which are usually generated by many different techniques.

2.1 CAPTURE METHODOLOGY INTRODUCTION

The purpose of this multi-domain design capture and analysis methodology is to provide a simple and robust means for partitioning the system design data into comprehensive yet comprehensible groupings. This multi-domain approach provides a means for avoiding the problems associated with capturing design information from a single design viewpoint, or even from multiple loosely coupled (or uncoupled) design viewpoints. These problems can be illustrated by the typical process of constructing a system requirements or design baseline. High level system requirements, which often combine different perspectives of the system and can be inconsistent, incomplete, ambiguous and sometimes incorrect, are commonly specified in documents established by committees . Ambiguous requirements can then be interpreted by the system design team (who also see the requirements from different perspectives) and redefined without knowledge or intent of the requirement's author. Gross inconsistencies or errors are generally identified early in the design review process; however, less obvious and subtle inconsistencies or incorrect requirements may not be identified until late in the system design or implementation, causing critical system shortfalls and costly redesign.

The methodology for system design capture and analysis described in this technical report addresses each of the key system design perspectives and provides a mechanism for constructing a consistent, structured, multi-domain representation of the design. The overall goal is to provide a method with the following capabilities which are critical to support efficient capture and analysis of large, complex, computer-based systems:

Scalability of the method to very large complex problems and management of system complexity and size through automation;

Unambiguous representation of system performance requirements and concept of operations providing a common means of communication between the sponsor, users, architects, analysts and engineers;

Separation of functional and implementation design issues and support for simulation and analysis of candidate functional designs and candidate resource implementations;

Cross-linking of design capture information in rigorous relationship definitions to avoid redundant data capture and to support intensive automated bookkeeping tasks;

Consistency checking and traceability across design capture elements and domains;

Generation of system documentation from information captured in the design database;

Applicability of the methodology to full-scale automation of system design capture and analysis process.

The central element of the multi-domain design capture and analysis methodology includes definitions of the multiple design domains or views which address the principal system design perspectives: (1) Informational, (2) Functional, (3) Behavioral, (4) Implementation, and (5) Environmental. The capture approach for each design domain or view share a common hierarchial structure which supports management of the magnitude and complexity associated with a large system design. Flat representations of complex system designs rapidly become unwieldy as the design detail unfolds. A hierarchial structure allows the system capture views to be represented at various levels of detail from a broad top level, which encompasses the breadth and scope of the system and its external interfaces, to very low levels which describe the details of a particular segment of the system design.

The system design capture views are a representational means for populating categorized repositories of design information that is retained throughout the system design process. Regardless of the specific process (e.g., waterfall, spiral, virtual prototyping, etc.), there are some similar activities which need to be performed (e.g., planning, specification, analysis, integration, test, etc.), and information exchange between activities is required. Design information that is provided from a common point of reference will guarantee the results of various analysis techniques to be consistent which, in turn, produce a better design.

In the forward engineering process, top-level system requirements are partitioned into appropriate capture views. These requirements are abstracted into design guidelines or rigorous constraints that system designers must follow. As the design is extended, further detailed information is added. These requirements are continuously traced through all design elements to insure the satisfaction of the design.

The capture views provide a framework for maintaining the design information at any level of detail; hence, the design can be evaluated at various levels of abstraction. However, the capture views do not necessarily provide the exact format which is required by a particular evaluation technique. Instead, they maintain, within their own format, a set of information that is

pertinent for most evaluation techniques. Information that is specific to a particular evaluation method is expected to be derived from the capture views. This prevents the overload of information on capture views yet maintains the consistency between evaluation models. During the construction and the analysis of the evaluation models, assumptions can be made; however, they must relate back to the capture views in conjunction with the evaluation results. Given that the evaluation results suggest many variations in the design, the capture views should be able to handle different design options and the rationale behind the selection of these options. The close loop between system design capture and system design evaluation, including optimization, simulation, and analytic analysis, allows systems engineers to gradually refine the design in many different aspects (e.g., performance, reliability, security, etc.).

Since the key issue in reengineering is the ability to understand the functionality, behavior, and implementation of the existing system, the design capture views framework naturally supports this process. Depending on the level of reengineering, information about the existing system can be specified in the appropriate capture views. Supporting various levels of detailed descriptions as well as multi-levels of design abstraction, the capture views provide a framework for the whole spectrum of reengineering. For example, for the reuse of existing software, the functionality of the existing code can be captured in the software architecture of the Implementation Capture View. This software architecture provides a better understanding for the software function of the existing code which can then be assessed for its use in new required applications. For the reengineering at system or subsystem levels, the functionality, behavior, and implementation of the existing subsystem (or system) can be captured by the Functional, Behavioral and Implementation Capture Views.

The capture views also provide guidance for system test and integration. Test strategies will be established for the entire system as well as individual critical elements within the design. However, one should not expect a detailed test procedure from the capture views. The intention here is to establish critical check points which are based on the requirements and design decisions such that test procedures can be defined properly. This will also allow test results to be traced easily to certain design selections and/or requirements to ensure the satisfaction of the design.

The five system capture views partition the system design into logical segments which correspond to key perspectives of the system design. These five capture views are distinct but related representations of key aspects of the system. They are summarized in Table 2-1.

The Informational Capture View captures a conceptual representation of the system under design in abstract terms. This view captures all components that make up the system and the interaction between these components. This allows for a description of the intended system concept of operations without implying or constraining the physical implementation of the system under design.

The Functional Capture View establishes the functional structure of the system. It specifies how the functions are decomposed and how the information is transformed through these functions. This provides a better understanding of the system's functional decomposition.

TABLE 2-1. SYSTEM CAPTURE VIEWS

DESIGN CAPTURE VIEW	VIEW OBJECTIVES	DESIGN ELEMENTS
Informational Capture View	■ Characterize system concept of operations ■ Represent system components in abstract terms	■ Entity-relationship diagrams ■ Attribute/method descriptions
Functional Capture View	■ Define system functions and decompositions ■ Specify data flow requirements	■ Function/data flow diagrams ■ Process specifications ■ Data dictionary
Behavioral Capture View	■ Define system critical paths ■ Specify system behavior characteristics	■ Control flow diagrams ■ System behavior threads
Implementation Capture View	■ Define the physical hardware, software, and human resources which make up the system ■ Specify system physical interconnectivity	■ Hardware, software, and human resource descriptions ■ Performance parameters and resource characteristics
Environmental Capture View	■ Establish conditions and events constraining system operations ■ Specify performance MOEs and conditions of measurement	■ Environmental conditions and event descriptions ■ System initial conditions ■ Measures of effectiveness

The Behavioral Capture View describes the system's attributes over time and describes the event or time-driven aspects of the system. This view allows for specification of the system's behavior at different times and under various conditions and situations.

The Implementation Capture View defines the physical hardware, software and human resources which comprise the system and its connectivity to external systems. A multi-level mapping scheme which relates the system functions (with associated data flows and behavior contained in the Functional and Behavioral Capture Views) to the resources in the Implementation Capture View is also established.

The Environmental Capture View captures the system under design from the user's or operator's point of view. This view describes the situation(s), environment(s), expected events and other factors that make up the conditions under which the system is operating. This includes the following: the initial state of the system under design; environmental conditions including acoustic, electromagnetic, and meteorological conditions; hostile threat platform types and locations (in a military application); operational constraints; likely strategic and tactical considerations and other pertinent items; and concept of operations. The measures of effectiveness (MOEs) which characterize system performance and establish the "mission success criteria" for the system are also specified together with the conditions under which the MOEs are measured. The Environmental Capture View also specifies the guidance and constraint of the environment which the design itself must face.

An attempt to address all of the issues associated with these capture views simultaneously or without a structured methodology is a multi-dimensional problem of a magnitude which exceeds the capacity of most, if not all, systems engineers. Each of these capture views provides

key information concerning particular aspects of the system under design. Taken individually the capture views allow the systems engineer to partition the design of a proposed or existing system into manageable parts. The five system capture views introduced in this section will be described in detail in Sections 3.1 through 3.5.

As systems increase in complexity and size, the capturing design methods have to expand extensively so that they can be able to capture different aspects of the system in a complete and systematic manner. Deciding what aspect of the system needs to be captured and to what level of detail is a very difficult task. Sometimes the same group of attributes can be used to completely capture one type of system, but for another type it becomes irrelevant or incomplete. Therefore, the ideal situation is to define the capturing method for all aspects of the system and, depending on the system requirements or on the design phases, emphasize or deemphasize certain system capture views.

2.2 SYSTEM DESIGN CAPTURE VIEW ANNOTATION

A design element is a partitionable component of the specification. It can be informational, functional, behavioral, implementation, and/or environmental and can describe function(s), behavior(s), data, object(s), relationship(s), physical implementation(s), etc. As part of the system design capture views, the systems engineer must not only capture information about how the design elements interrelate, but also have extensive information about the design elements themselves. Relationships between the various system design factors can be expressed formally so that effective trade-offs can be performed between the various design factors.

Design factors are attributes that are attached to design elements. Appendix A is an initial list of the design factors that are typically used in the design of mission critical systems. These design factors are non-functional in nature; they describe how the design component will operate, not what the design element will do. For example, in the sonar system, the detection function can be annotated with the throughput, response time, and availability factors in addition to the description of the detection functionality.

Typically, a design element related to a particular design factor may be annotated in three ways: (1) requirements/budgets, (2) predicted/expected, and (3) actual.

The first category is the required need for that design factor within the design element, either specified by the original requirements, or budgeted by the designer in order to meet an overall requirement. The second type of annotation describes what has been allocated by the systems engineer to the design element. This is realistically what the system design expects to be able to meet in the design. The allocated value of a design element should specify a better or equal performance than the required value. The third category describes actual measurements for the design factor's value. This is used in the validation of the system design. Actual measures at the "leaf" design elements can propagate to the other design elements, yielding the verification and validation of various portions of the design. Earlier in the design process, the information available (through rapid prototyping or existing candidate hardware/humanware characteristics) can be used to guide design decisions.

Additionally, the design elements should be annotated with a description of a design rationale which leads to the design of that particular element. This provides background for future modification or selection of alternative design options.

CHAPTER 3

ESSENTIAL SYSTEM DESIGN CAPTURE VIEWS

In order to design any type of system, systems engineers should be able to clearly understand the components that make up the system. Defining a proper method that can correctly capture different types of system components becomes a critical issue for the methodologist. The following description of the five system capture views and their supporting methods is an attempt to formally and completely capture the system under design. To illustrate the defined methods, the five capture views of the passive sonar example is documented in Appendix B. Readers should keep in mind that this is a working document which focusses on the capture methods, not on the design of the passive sonar system. Due to both the infantile methods and/or the automation support, some information in the example is inconsistent and, sometimes, incomplete. However, through this example, the usefulness of the capture views can be demonstrated.

3.1 INFORMATIONAL CAPTURE VIEW

The Informational Capture View captures all elements that make up the system and shows how they interact with each other. Components of the system are represented as abstract objects and are associated with each other through relationships. Depending on their performance, objects and relationships at the boundary of the system will also be studied to complete the Informational Capture View. This view provides a flexibility for the systems engineering design team to explore and communicate various design concepts with the customers in abstract terms without prematurely influencing the system implementation in hardware, software, and human operators, or in specific functionality. Even though the focus at this level of design is to narrow the design space and to gain a thorough understanding of the requirements, capture techniques at this level must be robust enough to satisfy the following needs: (1) easy to use and understand by non-domain specific personnel; (2) scalable to handle large, complex systems; and (3) flexible enough to represent different types of system components as well as the quality of these components.

3.1.1 Example Method

In the past, the information modeling concept has been used in the development of systems and has had some success. As systems change their size and mission, different or additional information modeling concepts are developed to support them. The object-oriented approach to analyze systems by Shlaer and Mellor[4] is one of the most well known methods. It uses abstract objects to represent real world components of the system and uses the entity

relationship symbols to model the connectivity of the objects. Shlaer and Mellor[4] believe that a large, flat graphical information model can carry enough information to represent any system.

Another object-oriented method which was introduced by James Rumbaugh, et al.[8] offer a better set of notations for modeling and design. A significant improvement of this method is the ability to represent the grouping of objects through the aggregate relationship. Using these notations, the complexity of objects can be unfolded to a certain level of detail, and the complicated relationship between objects can be represented.

Although the existing object-oriented methods were successfully used in the development of software, they are insufficient when applied at the system level, especially in handling the size and complexity of large Navy systems. The Naval Surface Warfare Center Method and Tools Working Group[6] adopted and modified the Shlaer and Mellor concept by introducing the hierarchical representation of objects. Following up on that work, this thrust (Real-Time System Design Capture and Analysis) enhanced the definition of objects and relationships. These changes allowed for objects and relationships to be represented at different detail levels. This representation is consistent with the information hiding theory that was known to be beneficial for the capture of large-size systems. The basic concept of the Hierarchical Information Model (HIM) is the incorporation of the hierarchy notion into the Shlaer and Mellor object-oriented approach. Following is the definition of objects and relationships that were used in the HIM.

3.1.1.1 <u>Object</u>. In general, an object is defined as an abstract representation of the real thing that makes up the system. Each object has a unique name and, depending on whether it is simple or complex, its characteristics will be captured by a list of attributes or by another set of objects.

All information that reflects the characteristic of an object or can be used to identify an object should be represented in the object attributes. Some system design factors that are listed in Appendix A are good examples of likely object attributes. Each attribute should only capture a single relevant piece of information to the object being defined and take on the values independently. Due to the complexity of the system and the level of detail that must be captured, many types of objects have been identified. To follow the example easily, readers should keep in mind that the capture must include objects that are within the scope of the design and also objects that are outside the design space but have direct impact on the design. In addition, the object's representation must allow for the distinction between "simple" and "complex" objects. A simple object implies that information on the component part of that object is not available and, vice versa, a complex object implies that the component part of that object is specified. Page B-7 summarizes the notations that were used in the HIM of the passive sonar example.

3.1.1.2 <u>Relationship</u>. A relationship shows how objects are associated to one another. To emphasize that the relationship does not signify data flow, it is represented by a straight line that ends by a white or black ball (see page B-7). The ball indicates the "inferior" side of the relationship. While a white ball indicates that the simultaneous existence of the inferior and superior objects are not required at all times, a black ball indicates that the existence of the inferior objects is dependent on the existence of the superior objects. Attached to each end of a relationship is a set of cardinal numbers which is used to indicate the number of relationships that

may exist simultaneously between the objects. The cardinal number is expressed as "min:max," where min is always less than or equal to max and can have a value as small as zero.[9]

Another type of relationship between objects is the subtype/supertype which enables the designer to model similarity between objects. An example of this relationship is illustrated in Figure 3-1. In the real world, merchant ships, combat ships, and cruise ships can be viewed as different objects. At some instant, they can be referred to as a same object called ship which carries a set of common attributes among them. Merchant, combat, and cruise ships are called subtype objects and will inherit all attributes of their supertype object ship. In addition, each subtype object will have its own additional attributes to represent itself. For example, all types of ships will have a serial number, size, and destination; but combat ships will have different types of weapons and missions as attributes to distinguish between themselves. Consequently, the subtype object inherits everything, without exception, from their supertype object.

FIGURE 3-1. EXAMPLE OF SUPERTYPE/SUBTYPE RELATIONSHIP

3.1.2 Example Capture

Section 2 in Appendix B (B-6 through B-17) illustrates the representation of the Implementation Capture View of the passive sonar example using HIM. At the context level, the whole system is represented by one object, the PASSIVE SONAR object. Using the above notations, the context level PASSIVE SONAR SYSTEM (see page B-9) can be expressed:

- PASSIVE SONAR is a "complex" object and internal to the design space;
- POSITIONING SYSTEM, MASTER CLOCK, CIC, and TARGET are objects that are outside the scope of the design but have direct impact on the design;
- TARGET is a "complex" object and also has three different subtypes: SUB, SUR, and FALSE;

Similarly, on page B-11, information about the SIGNAL PROCESSOR object can be read as follows:

- The SIGNAL PROCESSOR object is composed of APPLICATION PROCESSOR and ACOUSTIC PROCESSOR which are both internal to the design space;
- DISPLAY CONSOLE and ANALOG SIGNAL CONDITIONING EQUIPMENTS are objects that are within the system under design but are not part of the SIGNAL PROCESSOR;
- BEAMFORMER, NETWORK, and SYSTEM SUPPORT SERVICE are "complex" objects that are within the system under design and are not part of the SIGNAL PROCESSOR. More information about these objects are captured on other diagrams;
- CIC is a "simple" external object.

3.1.3 Discussion

Although the concept of HIM has been introduced, it is still immature and much effort is required to complete this work. Besides solving the complexity and scalability issues of the Information Modeling techniques, there is still a need to define the way to capture the qualification of the object's attributes.

Within our context, the HIM is only employed as a communication vehicle between customer and systems engineer. The flexibility and abstract nature of an object's description allows a customer, who has a better understanding of how the system must operate as a whole unit and under what type of environment, to specify the system's requirements and design limitations without constraining the actual design decision and implementation.

In the last decade, the object-oriented method was promoted extensively for software system design and development. It was used in various stages of the software design process including modeling, analysis, and design. Although advocates of the Object-Oriented Design (OOD) method claim that these techniques can also be applied to large-scale, complex system design, its success is still questionable. Several extensions to the original object-oriented concept have been defined to employ this method at the system design level: state machines are used to represent the object's behavior;[4] and objects are used to represent system resources at a high level of abstraction.[10]

Nonetheless, the OOD notation tends to force early decisions concerning system implementation because, as objects are identified, they directly imply a specific implementation. In large and complex system design, a multi-domain design synthesis and analysis approach offers multiple opportunities for design trade-offs and design options analysis. This is of particular

significance in large-scale systems where multiple implementation options exist for a given set of functional requirements.

3.2 FUNCTIONAL CAPTURE VIEW

In designing large-sized and complex systems, engineers do not have the luxury to build and test the complete system as in designing small-sized systems. It is important to study and estimate correctly what the system will do so that errors can be prevented early in the development process. System functions need to be organized in a clear, systematic, and hierarchical manner so that engineers can see how the functions are arranged, how they interact with each other, and what other options are available in partitioning these functions.

The Functional Capture View is the representation of the system structure: how functions in the system are decomposed and how they interact with each other. There are many existing notations to represent system functions including data flow diagrams (DFD) and functional flow block diagrams (FFBD). For this discussion, the DFD is selected as an example of a method to represent the Functional Capture View.

3.2.1 Example Method

The capture of the Functional Capture View is based on the Structured Analysis method.[1] The DFD (or Context Diagram) is used to represent the system functions and the data flow between them. The graphical and hierarchical representation of these diagrams allows systems engineers to analyze the functional construction of the system. Some extension (i.e., the specification of nonfunctional attributes and design rationale) to the above method is incorporated to enhance the capturing technique.

3.2.1.1 Context Diagram. Context Diagram is a special case of the DFD. It represents the top level of the design where the boundary of the system under design is defined. At this level the whole system is represented by one process; any other outside elements that interact with the system will be represented by terminators. Using the current method, terminators only communicate directly with the system. Research shows that if the interfaces between terminators affect the system's performance, then they have to be captured and studied at certain levels of detail.

A terminator is represented by a square box and its name should reflect the function or information that it carries. Data flow is still used to show the flow of information both from and to the terminators. B-19 illustrates the Context Diagram of the passive sonar system. The boundary of the Passive Sonar System functionality is represented by one process named PASSIVE_SONAR_SYSTEM_PERFORMANCE. External functionalities (or objects) that impact the operation of this system include NAVIGATION, TARGET, and CIC.

3.2.1.2 Data Flow Diagram (DFD). The DFD is a network of processes and data flows. Requirements are partitioned into processes (or functions), and the information that passes

between processes is represented by data flows. The DFD includes the following components: processes, process specifications (P-Specs), data flows, data flow specifications, and data stores. B-20 shows an example of a DFD which includes five processes, data flows, and control flow.

3.2.1.2.1 PROCESSES/FUNCTIONS. The process represents the transformation of information using incoming data to produce outgoing data. Each process is represented by a circle and has a unique name. The process name usually corresponds to the function that it performs such that engineers can have a general idea of what the system is doing.

To extend the transformation in more detail, engineers decompose the process into many different processes. There is no limitation on the level of decomposition; however, all lower levels of decomposition of a process must reflect the same transformation. Also, not all processes have the same level of decomposition. Functions that are more complex require extra partitioning levels. Page B-20 shows the next level of decomposition of the PASSIVE_SONAR_SYSTEM_PERFORMANCE which includes five processes: STABILIZATION, SIGNAL_PROCESSING, TIME_SYNCHRONIZATION, DATA_FORMATTING AND OPTION SELECTION, ACOUSTIC_ENERGY_CONVERSION.

3.2.1.2.2 PROCESS SPECIFICATIONS. In the Structured Analysis method, each process specification or P-Spec is used to describe the functionality of a process at its lowest level of decomposition. P-Specs exist in either textual or mathematical format to specify how processes operate.

The design capture views methodology has identified that in order to provide sufficient information for assessment at all levels of decomposition, each process should have an associated P-Spec. Also, these P-Specs must include the appropriate design factors list to describe the quality of the required system's function, the design rationale, and the suggested compatibility. An example of P-Spec is included in Appendix B (B-27).

3.2.1.2.3 DATA FLOWS. Data flows represent the flow of information between processes. A data flow is represented by an arrow going from one process to another. Its name is unique and reflects the information that it carries. Like process, each data flow can be a composite data flow or a primitive data flow. A composite data flow is a combination of other composite data flows, or primitive data flows, or both.

When the processes are decomposed, their associated data flows will have to be balanced at different levels of decomposition. Figure 3-2 graphically demonstrates the decomposition of processes and their associated data flows.

3.2.1.2.4 DATA FLOW SPECIFICATIONS. Like P-Spec, a data flow specification is used to describe the data flow's attributes. Format of the data flow specification is similar to P-Spec. Appendix B (B-64) includes an example of the data flow specification.

FIGURE 3-2. EXAMPLE OF PROCESSES AND DATA FLOWS DECOMPOSITION

3.2.1.2.5 DATA STORES. Data stores represent a collection of information that is often reused by processes. Information in the store will remain the same until it is updated.

3.2.2 Example Capture

Section 3 of Appendix B (B-18 through B-92) illustrates the data/control flow diagrams and their associate information of the passive sonar example. This example shows a few top levels function of decomposition of the passive sonar system. Example format for P-Specs, data flow specifications, and the utilization of System Design Factors (SDFs) are also included; however, the values are not completely/correctly provided.

3.2.3 <u>Discussion</u>

A previous shortfall of the capture method for the Functional Capture View did not address the non-functional attributes of the system, such as dependability, security, and reliability, which are critical to the system's performance. Detailed information of the non-functional attributes are now captured as design factors, described in Appendix A, and allow systems engineers to tag this information directly to the functions at any level of decomposition. In the passive sonar example, an attempt to generate P-Spec for all processes at all levels of decomposition has been conducted. This will provide sufficient information about the functional design such that it can be assessed at any level of decomposition. However, it also implies that a formal set of rules must be defined for the relationships among the non-functional attributes especially for their hierarchical breakdown.

In addition, current automation support has limited capabilities which make it inefficient when developing large models. For example, two problems were discovered with the Teamwork CASE tool's automation of the Yourdon-Demarco method. First, Teamwork does not allow for automatic replication of identical parts of a system. When simulating a large system, it becomes necessary to replicate parts of the system, and therefore, it is important that parts of the model can be replicated efficiently. The second shortfall is that it does not offer automatic analysis. The system can be analyzed based on various requirements. Each analysis technique requires certain types of input and usually has its own notation. Although the required information exists in the functional description, the question is how to automatically translate it into the analysis format and still guarantee the completeness and correctness of the design.

3.3 BEHAVIORAL CAPTURE VIEW

Real-time systems dynamically change over time and under different conditions. In order to correctly predict the performance of the system, it is essential for systems engineers to capture the causes that make the system change its mode of operation and describe how the system behaves afterward. A comprehensive description of system behavior in response to the environment in which it will be operated allows various types of analysis to be performed; hence, it provides the means to verify the feasibility or correctness of the design in meeting its requirements.

Since the behavior of a system can be examined from different points of view, its representations can be diverse. Existing system behavior representation techniques tend to address a particular aspect of the system behavior, and transition between these representations has never been clearly defined. The challenge of the Behavioral Capture View is to provide both the techniques to represent the total behavior of the system at various levels of details and the transition (both forward and backward) from design capture to various types of analysis.

3.3.1 Example Method

The initial approach for the capture and analysis of system behavior provides a mechanism for representing key aspects of system behavior at various points in the design process, for relating system behavior to other design capture views, and for checking the consistency of the various behavior representations. This approach does not exclude the use of existing capture methods for system behavior; instead, it provides a structured framework for the analysis of system behavior through different phases of the design process.

This approach partitions system behavior into three elements: (1) top level system behavior as observed externally (i.e., system as a black box), (2) real-time behavior of the system functional design options (i.e., control, timing, and synchronization aspects of the system functions and data without regard to specific implementation in resources), and (3) real-time behavior of system implementation design options (i.e., response time, delays, queue lengths, and other aspects of the system behavior as embodied in the hardware, software, and human resources and resource architectures).

3.3.1.1 Top Level System Behavior. The top level system behavior is characterized by response of the system to external stimuli/events/conditions and to internally generated events/conditions. At this level, a system is treated as a black box and the system behavior is described as the required timing, synchronization, and response of the system in relationship to the external environment in which it will operate. Figure 3-3 illustrates the concept of top level system behavior. The top portion of Figure 3-3 illustrates the simplest case of top level system behavior where the system generates an action or event based upon a single external event or an internally generated event. The lower portion of Figure 3-3 illustrates a more complex case where a chain of multiple external and/or internal events are combined in a sequence with system generated responses. In each of these cases, the top level system behavior capture method must describe each external condition/event and each system response as well as the relationship of those events over time (e.g., the sequence) to adequately capture the behavior.

Top level system behavior is closely related to the descriptions of scenarios and test conditions captured in the Environmental Capture View (see section 3.5). Many top level system behaviors may be described in terms of a response to an external event. These external events are captured as part of scenario descriptions. A strong coupling between the descriptions of top level system behavior within the Behavior Capture View and the scenario descriptions within the Environmental Capture View will need to be explored.

FIGURE 3-3. TOP LEVEL SYSTEM BEHAVIOR

3.3.1.2 Functional Behavior. The functional behavior describes the real-time behavior of the system functional design in terms of the control, timing, and synchronization aspects of the system functions and data (without regard to specific implementation in resources). This element of system behavior is often necessary to completely specify the real-time aspects of the system logical design (describes "what" the system will do in terms of functions and data flow, including timing and synchronization, without regard to a resource specific implementation). Specification of data flow alone is not always sufficient to uniquely determine the timing and synchronization aspects of the system functions. This is particularly true when trading off sequential versus pipeline timing of system functions or operations as illustrated in Figure 3-4. The two examples in Figure 3-4 have identical functions and data flows; however, the control structures are different. The control structure on the left implements sequential operations wherein all functions in the chain operate in sequence on a given initial input before the next input is processed through the chain. The control structure on the right implements pipeline operations wherein each function in the chain operates as soon as input from the previous function is available allowing multiple data elements to be processed through the chain simultaneously.

FIGURE 3-4. SEQUENTIAL VERSUS PIPELINE TIMING OF SYSTEM FUNCTIONS

The system functional behavior is strongly related to the top level system behavior described above. The top level system behavior considers the system as a black box. The stimulus/response description at this level naturally serves as endpoints in tracing system functional behavior (which views the system as a white box) through the system as illustrated in Figure 3-5. The capture method for representing system functional behavior provides a mechanism for relating to top level system behavior and for tracing behavior through function/data flow threads in the functional design.

FIGURE 3-5. SYSTEM FUNCTIONAL BEHAVIOR CAPTURE

3.3.1.3 Implementation Behavior. The implementation behavior is the real-time behavior of the system's implementation (or resource) design in terms of the service time, delays, queue lengths, and other aspects of the system real-time behavior as embodied in the hardware, software and human resources and resource architectures. The behavior of system hardware and software architectures is most often represented as selected testing threads through the system design. Each component in the path of a given thread contributes to the overall behavior of the system in terms of the service time, delays, etc. This approach tends to isolate selected behavior issues for analysis and can often mask the effects of multiple concurrent threads (or contention for resources across threads). An extension of this approach (which represents individual threads through an implementation) is to represent the real-time behavior of individual components and then to combine those components into a total system architecture representation. This extended approach still allows threads to be traced through the system implementation design and has the added capability to represent and analyze multiple concurrent system implementation threads.

As with the system functional behavior threads, system implementation behavior threads are strongly related to the top level system behavior described above. The stimulus/response description of the top level system behavior serves as endpoints in tracing system implementation behavior as illustrated in Figure 3-6. The capture method for representing system implementation behavior must provide a mechanism for relating to top level system behavior and for tracing behavior through hardware/software/human operator threads in the implementation design.

FIGURE 3-6. SYSTEM IMPLEMENTATION BEHAVIOR CAPTURE

3.3.2 Example Capture

An example of top level system behavior is the response time of the system to a user action such as a data base query. The initial external event which triggers this behavior may be represented by the entry query of a system operator. The system response is represented by displaying the necessary information on the operator's display. The time associated with this top level system behavior of "system response time to data base query" may include a variety of components or different values. A required or budgeted value may be derived from the system top level requirements. A predicted or simulated value may be generated from an analysis of the system under design and a measured or observed value may be recorded from an actual test. A mechanism must be provided to capture and relate the spectrum of values which may be generated. Both text-based and graphic-based mechanisms are being considered as candidates for capturing a description of top level system behavior. An innovative graphic story boarding technique which uses icons and symbols to represent various events and their relationship in time is a leading option. Another candidate capture technique for representing top level system behavior includes the state-based method.[11] For the design of software system, several methods were advocated to support the capture of system behavior. These methods include Modechart,[12] Communicating Sequential Processes,[13] and Z Formal Specification Language.[14]

Candidate methods for capturing system functional behavior include the use of control flow diagrams and/or state transition diagrams to specify the sequence of operation for a set of system functions. Candidate capture techniques for representing system functional behavior in the form of control diagrams include RTSA,[1] HGSDCSM,[15] and SYSREM.[16]

Existing methods are not designed to support the capture of system implementation real-time behavior. Capture capabilities which are provided by some analysis/simulation techniques can be used for this purpose. However, they are not effective. These methods allow systems engineers to mimic some behavior attributes of the system's components but they do not provide a complete picture of the system resource architectures and their associate behaviors. Among these methods, Petri Net based methods are commonly used.[17]

3.3.3 Discussions

The collective analysis results of the three levels of system behavior provide a solid foundation for the understanding of the design. At each level, the results of the analysis support systems engineers in focussing their design choice to satisfy the requirements. Hence, it is important that the transition between the three defined levels of system behavior is established. An efficient transition capability must support the trace of the capture information between different levels of representation as well as the progression from design capture to design evaluation.

Successful employment of an advanced system behavior capture and analysis methodology in supporting a complex system design is largely a function of the degree of mechanization which can be achieved. The size and complexity of large-scale advanced computer systems render manual application of any design process or method unusable. Considerable potential benefits can

be gained from automation which supports a disciplined structured capture of the initial iteration of a system design and subsequent editing of that capture. Further significant efficiencies can be gained through automated consistency and completeness checking within the behavior capture domain and between the other system capture views which represent the captured design. However, in light of the overwhelming systems engineering task represented by analysis of an advanced complex system design, the most significant productivity gains may be in automated support for design simulation and analysis within an integrated and highly automated design capture and analysis environment.

3.4 IMPLEMENTATION CAPTURE VIEW

An important aspect of the Implementation Capture View is the method employed for representing system resources and resource architectures which embody a given system implementation option. The term "resource" is used here to represent the components (e.g., hardware units, software, or human operators) that comprise the system's implementation design, and the term "architecture" represents how those system components are combined (e.g., interconnection of hardware or software).

Once the system's functions and behaviors are identified, to avoid massive failure in the design phase, it is important to understand the resources that will be used to perform these functions. The capability to capture and analyze different resources to achieve high performance at low cost in the early stage of the design cycle will yield maximum payoffs in the production and maintenance phases. The analysis of the resources also reduces the complication in upgrading systems when better technology is introduced. In reality, most resources already exist and their performances are documented informally. If the resource representations are formalized, then engineers can study the trade-offs of one resource versus the other, and can identify which resources can be modified or will need to be built for the system under design. A robust, flexible mechanism is required for characterizing these system resources, since it must support optimization of the system design with respect to competing system requirements such as performance, reliability, cost, and security, as well as trading off alternate hardware architectures and software partitioning early in the design process.

3.4.1 Example Method

The resource capture methodology supports the characterization of complex computer-based system resources across the broad spectrum of resource types and at various levels of design detail.

3.4.1.1 Objective and Requirements. The objective of the resource capture methodology is to provide a capability to represent a wide spectrum of resource descriptions for complex systems designs. Depending where it is in the design process the resource descriptions may vary from very specific to generic, from abstract to detailed, and from simple to complex. In a top down design process, the design implementation progresses over time from high level abstractions of system resources to increasing levels of detailed hardware, software and human operator

representations. Analysis and simulation of a particular design option may be required at various points in the design process to address trade-offs or demonstrate compliance with key system requirements.

The resource capture methodology must therefore provide a robust, flexible method for representing resources of many different types and levels of abstraction. Figure 3-7 depicts four levels of abstraction for representing the spectrum from very highly aggregated resources down to resources at the sub-component level. (Four levels are picked for illustration purposes only; more or fewer levels may be required to support the design capture and analysis needs of a given project.) Early in a top down design process, the system resources may be represented as large, highly aggregated resources (e.g., subsystems) and the system under design is represented as a combination of interconnected complex subsystems. These subsystem resources are decomposed at the next level of design detail and are represented as aggregations of hardware and software components. At the next level, components are individually represented including software configuration items (with their associated mapping to tasks in real-time system execution) and hardware units with associated Lowest Replaceable Units (LRUs). Typically the systems engineering activities do not extend below this level of detail; however, this resource capture methodology should support linkage with hardware and software engineering methods and tools and thus should ultimately support this entire spectrum.

FIGURE 3-7. SPECTRUM OF RESOURCE DESCRIPTIONS

The resource capture methodology must also provide a mechanism for managing traceability in the characterization of resources across these various levels of design detail. This includes establishing equivalency relationships between a given subsystem and its equivalent representation as an interconnected group of component parts. One or more sets of formal rules and consistency checking paradigms may be required to support semi-automated linking of

resource representations at a higher level of abstraction to resource representations at the next lower level of design detail. This capability is necessary to support decomposing resource descriptions in a top down design scenario or recomposing them in a re-engineering or bottom-up design scenario. Techniques must also be provided to manage multiple concurrent or alternate resource and design options for various subsystems or for the entire system under design.

3.4.1.2 Resource Capture Methodology. A mechanism for efficiently representing a large number of resource types must be provided as well as a flexible, robust approach to portray a diverse spectrum of resource characteristics and attributes. Combinations for various resource characterization techniques must be supported including singled valued parameters (e.g., weight, size), multi-valued parameters (e.g., reliability as a function of temperature), complex models (e.g. algorithms), as well as less formal English text or graphic depictions. A mechanism to generate system resource architectures also needs to be furnished for the understanding and evaluating of the design. Following the Shlaer and Mellor notation with cardinality annotated for each relationship, Figure 3-8 summarizes a mechanism that supports the generation of system resource architectures. This concept has been implemented in the Resource Capture Prototype as a proof-of-concept.[18] The central elements of the prototype include the resource type class hierarchy, the resource, the system/subsystem, and the resource library. Additional objects are included to portray the essential external relationships of the Implementation Capture View.

FIGURE 3-8. RESOURCE CAPTURE CONCEPT IM

3.4.1.2.1 RESOURCE TYPE. By leveraging the inheritance and polymorphism characteristic of the Object-Oriented method, the resource type can be organized in a manner which efficiently represents a large number of resources. This class of resource types exists to allow efficiencies in representing multiple instantiations of similar objects, and to leverage properties of inheritance. Each resource type in the class hierarchy is characterized by attributes, methods, and interfaces and can be viewed as a template from which a particular resource object can be instantiated. An example of a resource type hierarchy is illustrated in Figure 3-9. A generic resource is classified into five subtypes: Hardware, Software, Human, Hybrid, and Document. Each of these resource types is further categorized into other subtypes (e.g., Software is arranged into five subtypes: Data Storage, Program, Message, Port, and Channel). Attributes associated with each type include those that are specified at their own level and attributes that are specified for all of its supertype levels.

3.4.1.2.2 RESOURCE. Resource representations can be instantiated from the class hierarchy by specifying their particular attributes, methods, and interfaces which signify the uniqueness of a specific resource type. Multiple resource objects can be instantiated from any of the resource types. Any resource object instantiated from a given type has, by definition, common attributes (though possible different attribute values), methods, and interfaces and therefore are interchangeable. This polymorphic characteristic of resource objects instantiated from a common resource class is a powerful feature allowing systems engineers to easily substitute alternative resource objects (given they are instantiated from a common type) in candidate architectures. Figure 3-10 illustrates an example of a hardware resource object with an abbreviated list of attributes.

3.4.1.2.3 SYSTEM/SUBSYSTEM. System/Subsystem is the interconnection of instantiated resource types (through their interfaces) using a building block approach. The attributes and methods associated with the interfaces of each resource object establish the means for determining which interfaces are compatible and for defining the topology and characteristics of links between resources. An arbitrary group of resources can be connected via their respective interfaces to create an architecture (e.g., system or subsystem) at various levels of abstraction. This group can then be associated with a single resource object at the next higher level of abstraction. This grouping of resources provides a mechanism for organizing system complexity through information hiding which supports top down or bottom up design approaches. The concept that supports the generation of the resource architectures (e.g., systems or subsystems) is summarized in Figure 3-11.

3.4.1.2.4 RESOURCE LIBRARY. The resource library is a repository for descriptions of instantiated resource components, subsystems, and systems. It also is a place holder for the system resource information that will be accessed by analysis tools to evaluate the design. Each element in the library includes an excessive amount of information and has multiple related versions. Efficient tracing and version controlling techniques are incorporated in the management feature of the resource library.

FIGURE 3-9. EXAMPLE OF RESOURCE TYPE HIERARCHY

FIGURE 3-10. EXAMPLE OF A RESOURCE OBJECT

FIGURE 3-11. RESOURCE INSTANTIATION AND AGGREGATION CONCEPT

3.4.1.2.5 REQUIREMENT TRACEABILITY MAP. The top level system requirements typically include specific implementation requirements and constraints which can be associated directly to the applicable resources captured in the resource model. High level design guidance and constraints (i.e., specific external interface requirements and protocols, use of standard computer hardware and software development standards, manning constraints, etc.) which are captured in the top level system requirements must be linked to the specific elements of the resource model that address them. In general, each implementation constraint will impact more than one individual resource such as a requirement for Mil-Spec computer hardware or the use of Ada in developing new software.

3.4.1.2.6 FUNCTION/BEHAVIOR ALLOCATION MAPPING. Each function and data flow in the Functional Capture View and each element of the Behavior Capture View has a strong association with one or more resources in the resource model. The function/behavior-resource mapping allows the systems engineers to audit the completeness of his design concerns/trade-offs. For example, the resource to which a function or data flow is mapped must have the necessary characteristics and capacity to perform that function or to achieve the data flow connectivity.

3.4.1.2.7 ENVIRONMENT. The information captured in the Environmental Capture View such as descriptions of the operational scenarios, external interfaces, and environmental conditions is needed to support simulation and analysis of the resource model.

3.4.2 Example Capture

Section 4 of Appendix B (B93-B156) illustrates the Implementation Capture View of the passive sonar example. It includes hardware, software, human resource architectures, and the descriptions of each resources. A spreadsheet describing the allocation of functions to resources is also included.

3.4.3 Discussion

Successful employment of a resource capture and analysis methodology in supporting a complex system design is largely a function of the degree of mechanization which can be achieved. The size and complexity of large-scale systems render manual application of any detailed resource capture method unusable. Considerable potential benefits can be gained from a highly automated design capture environment supporting a disciplined structured capture of resource descriptions which can be employed for design analysis, simulation, and optimization activities.

The above resource capture approach offers a robust, flexible mechanism for characterizing system resources including hardware, software, and human operators. In addition, it supports optimization of the system design in achieving competing system requirements such as performance, reliability, cost, and security, as well as trading off alternate hardware architectures and software partitioning early in the design process. A preliminary version of a prototype, the

Resource Capture Prototype (RCP), that implemented this concept has been released.[18] Further refinement of that product is currently planned.

3.5 ENVIRONMENTAL CAPTURE VIEW

The Environmental Capture View includes a representation of the outside world that will interact with the system under design. The intention here is not to model something that is not included in the system but to study the external interface of the system such that any real situation that may affect its performance is taken into consideration. The current approach is to extend the other four capture views beyond the boundary of the system under design and capture other environmental information that will affect the operation and the development of the system. It is important to understand that external information is only required for the analysis of the system under design; therefore, only external information which is relevant to that system is captured. The Environmental Capture View also includes information that impacts the design process itself.

The Environmental Capture View is defined as the structured representation and organization of the following types of information: operational scenarios, concept of operations, environmental conditions, external systems and interfaces, system test strategies, maintenance and logistics considerations, and other external factors which impact the system under design. The information captured in the Environmental Capture View addresses important issues of a system under design but may not typically be included in the design itself.

3.5.1 Example Method

Historically, the Environmental Capture View has been described in an ad hoc manner through test plans, context diagrams, simulation/stimulation (sim/stim) descriptions, and system modeling/simulation scenario descriptions. Since the capturing methods of this view are very immature, the current approach includes the following: (1) identify the appropriate information that needs to be captured and (2) generate the proper document to capture the selected information. The Environmental Capture View Approach report discusses this approach in more detail.[19] More formal representation techniques for this view will be defined in the future.

The key elements of the Environmental Capture View are grouped into two principal categories: (1) the operational environment and (2) the support environment.

3.5.1.1 The Operational Environment. The operational environment describes the situations and conditions under which the system operates. This includes the system concept of operation, the external system interfaces and external system, the environmental conditions, system test strategies, and the operational scenarios.

3.5.1.1.1 SYSTEM CONCEPT-OF-OPERATION. System concept-of-operation describes the proposed approach for operation of the system under design from the perspective of the customer or ultimate user. It specifies the philosophy and approach behind the operations of the system under design. During the early phases of the system development process, its

development applies to a broad spectrum of possible system implementations and is not constrained by a specific candidate system design, personnel manning plan, hardware/software implementation or existing system operator-machine interfaces. The focus is on describing the concept for mission execution using the system at a high level without regard to a specific physical system implementation. As the system design is developed over time, the level of detail captured in the concept of operations can then be refined to address design and implementation of specific features.

3.5.1.1.2 EXTERNAL SYSTEMS INTERFACES AND EXTERNAL SYSTEMS.

External systems interfaces and external systems identify and describe the external systems which have some relationship to the system under design and the external interfaces from these external systems to the system under design. The external interfaces between the system under design and the natural environment (e.g., temperature and pressure transducers) are also described. The principal purpose in describing the external systems is to provide a means for simulating their behavior and modeling their interfaces to the system under design. This information is necessary to create an external model which can be used for modeling the performance of candidate design implementations during early design phases or for interface stimulation during testing of the system.

This element of the Environmental Capture View also serves to summarize the system boundaries and increases understanding of how the system under design fits into the overall architecture and organization of which the system is a part. It does not address the internal structure of the system, but serves to define the system's interfaces and relationships to other systems and activities which are considered outside the scope of the system under design. It also describes and defines the objects external to the system and the behavior of those objects in relationship to the system under design.

3.5.1.1.3 ENVIRONMENTAL CONDITIONS.

Environmental conditions address mechanical (e.g., space and weight allowance) and service (e.g., electrical power and cooling availability) features of the location where the system under design is intended to be installed. The environmental conditions also include the circumstances under which the system must operate, such as geographic, meteorological, electromagnetic, and acoustic conditions. Some of these environmental data are typically archived in technical papers and journals which do not have a common format and are not organized for easy access and retrieval. Much of the data is available in hard copy (i.e., paper) form only with some electronic formats available (e.g., ocean acoustic environmental data). The purpose of the Environmental Capture View is to organize and capture this information in a manner which is useful to the systems engineer in designing large, complex systems and is not intended to duplicate the existing and ongoing efforts to collect and document environmental data.

3.5.1.1.4 SYSTEM TEST STRATEGIES.

System test strategies include system performance metrics, critical testing issues, and the overall concept for system testing to ensure that the system under development is compliant with the top level system requirements. The system test strategy can typically impose design constraints which must be identified early in the design process to avoid the potentially high cost of back-fitting test capabilities in the late stages of system integration and testing.

3.5.1.1.5 OPERATIONAL SCENARIOS. Operational scenarios describe situations and sequences of external events which are expected to be encountered by the system under design in both near-term and far-term over the system's projected life. The operational scenarios establish the bounds on the spectrum of possible scenarios, identify key scenarios which represent the most likely and most stressing cases, and describe selected key scenarios in detail. These key scenarios are captured in enough detail to establish test cases for system performance simulation and analysis.

3.5.1.2 The Support Environment. The support environment describes the information that is related to the post-design process and the supporting issues in the design process. Following is the information that must be captured in this section:

3.5.1.2.1 DEVELOPMENT, MANUFACTURE, MAINTENANCE, AND LOGISTICS CONSIDERATIONS. The development and manufacture plan for one system versus a number of identical systems is considerably different, and advance planning can reduce the cost significantly. Also, the maintenance plan for the developed system (or systems) can vary depending on the system's life expectation, the availability of the spare components (or subsystems), etc. In practice, system designers have a tendency to overlook these issues in the design process. This can increase complexity and cost in the production and maintenance phases.

3.5.1.2.2 SYSTEM ENGINEERING PROCESS, TOOLS, AND ENVIRONMENT. In dealing with a large-sized and complex system, it is important for managers or systems engineers to employ an engineering process and supporting tools at various levels of the process, to minimize the inconsistency or misunderstanding between different working groups. The environment under which the system will be developed also contributes to the success or failure of the project.

3.5.2 Example Capture

Section 5 of Appendix B (B-157 through B-164) illustrates some elements of the Environmental Capture View of the passive sonar example. Detailed information on this capture view is documented in reference.[19]

3.5.3 Discussions

The Environmental Capture View methodology establishes formal capture techniques for the environmental and external factors which can have considerable impact on the system under design. The potentially large amount of data which is included in the Environmental Capture View and the complexity of the relationships between the various elements leads to the conclusion that these capture techniques must be automated if they are to prove useful to the systems engineering team.

Further refinement of the techniques for characterization, selection, and description of operational scenarios, description of system concept of operations, capture of external systems and interfaces, environmental conditions, and system test strategies is required to: (1) address the employment of graphical and automated design capture and simulation methods, (2) enhance compatibility of the Environmental Capture View with other design views, and (3) provide a mechanism for traceability of system requirements and design rationale.

CHAPTER 4

DESIGN CAPTURE VIEWS AND THEIR INTERFACES

An attempt to address all of the issues associated with the design capture views simultaneously or without a structured methodology is a multi-dimensional problem of a magnitude which exceeds the capacity of most, if not all, systems engineers. Each of these views provides key information concerning particular aspects of the computer system under design. Taken individually the views allow the systems engineer to partition the design and analysis of a proposed or existing system into manageable parts.

The usefulness of the capture views cannot be clearly understood without the comprehension of their position in the entire development process. The relationships between the capture views and the key external entities which impact their constructions provide a better picture for the utilization of these views.

4.1 CAPTURE VIEWS AND THEIR EXTERNAL ENTITIES

As shown in Figure 4-1, the system design capture views exchange design information with five external entities. These entities impact the design process either by limiting the options which can be examined or by providing information which is required to fully capture the design. The following section discusses these relationships in terms of their effect on the design capture process (relationships among the external entities are not addressed, since their effect on the design capture process is exerted through the direct relationships with the external entities themselves).

4.1.1 System Requirements

Requirements are statements of need, characteristics of need, or conditions or capabilities that must be met or possessed by the system or its components to satisfy a contract, standard, specification, or other formally imposed necessity. In the context of the multi-view system design capture and analysis approach, the system requirements serve as a top-level description of the minimum standards which the system must meet in order to be accepted by the requiring organization. For this reason, the requirements entity (as embodied by such documents as Top Level Warfighting Requirements, System/Segment Specifications, or Mission Needs Statements) provides requirements to the design capture views. Each of the design capture views obtains requirements which define the necessary characteristics of the system in that view. Thus,

FIGURE 4-1. DESIGN CAPTURE VIEWS AND THEIR EXTERNAL ENTITIES

external conditions under which the system must operate (such as combat-induced phenomena) pertain to the Environmental Capture View; behavioral requirements (such as real-time performance criteria) pertain to the Behavioral Capture View; functional requirements (such as a listing of the functions which the system must perform) are provided to the Functional Capture View; and requirements which pertain to the system as a whole (such as generalized mission statements or objectives) are provided to the Information Capture View.

In return, the system designer must be able to trace each system design decision made in the capture process back to its underlying requirement. This tracing ability is typically implemented through the tracing of information in design documents back to its source in top-level requirements documentation. As defined in DoD-STD-2167-A, traceability has five elements:

- The [design] document in question contains or implements all applicable stipulations of the predecessor document;
- A given term, acronym, or abbreviation means the same thing in all documents;
- A given item or concept is referred to by the same name or description in all documents;

- All material in a successor document has its basis in its predecessor document, that is, no untraceable material has been introduced;
- The documents do not contradict one another.

Constraints are the other category of specification which flow from the System Requirements entity to the design capture views. Constraints are defined as restrictions which limit the set of acceptable system designs. Constraints, like requirements, pertain to specific design capture views. Thus, environmental constraints bound the design space for the system by specifying operating conditions; for example, a system which operates when portions are exposed to ocean depths greater than 200 feet would have limited options regarding those components. Similarly, implementation constraints limit the set of possible designs by reducing the range of hardware, software or human operator resources which can be used. For example, a system's software may have to be developed in ADA, and the functions must fit within acceptable limits on an AN/UYK-43 computer. These specific implementation constraints eliminate a wide range of potential design solutions which conform to the functional, behavioral, logical, and environmental requirements for the system as a whole. Finally, logical/informational constraints limit the possible set of informational designs of the system; for example, the requirements entity may specify a pipeline architecture, thus narrowing the range of logical design options available to the system design team.

4.1.2 System Design Evaluation

System Design Evaluation is the process of analyzing and optimizing the system design, or portions of the design, in comparison with the needs of the requiring organization. Specifically, it is the set of procedures and processes which the system design team and the requiring organization perform to determine the degree of acceptability and improvement strategies for a candidate design. This process ensures the following: 1) the design is complete and consistent; 2) the design is iteratively optimized until a feasible or optimal design is defined; and 3) the design can be partially implemented, through such means as rapid prototyping, on which tests can be performed to assess the acceptability of the candidate design. (These tests fall under the Test entity described below.) In order for the design to be evaluated, the design capture views must provide a detailed design description which contains the information resident in all five design capture views. This description must include not only the specifics of the candidate design itself, but also the environmental factors captured in the Environmental Capture View (such as operating conditions and scenarios) which support the evaluations of the design.

As a result of the analysis and optimization performed in the System Design Evaluation entity, a set of alternate design options are returned to the design capture views. These suggested options support changes made to the design information captured across the five capture views throughout the entire design capture process. For example, a design which fails to meet specified real-time behavior criteria might require the modification of its control structure; this modification and its supporting rationale, can be returned from the evaluation process to the Behavioral Capture View, where the change, with regard to the behavioral design of the system, can be made. Once changed, the design can then be reevaluated; this process is repeated until a feasible or optimized design is achieved. In addition, the design information contained in the

design capture views can be annotated with evaluation results, such as simulation performance, which justifies particular design decisions. This annotation supports the traceability of the design, since the origin of all modifications performed as a result of such evaluation can be identified.

4.1.3 System Test

System Test contains the formal process of preparation, conduct, and analysis of verification testing for the system and its subcomponents. This testing ensures that the candidate design meets all requirements before system development begins. Unlike the System Design Evaluation entity, in which emphasis is placed on determining how to incrementally improve the design, the System Test entity focuses on formal qualification of the candidate design *vis a vis* its requirements. Captured in this entity are all descriptions of the test environment, including hardware, software, and human resources needed. This entity also includes schedules for the tests themselves; the kinds and extent of formal qualification tests required (for example, stress, timing, or processor utilization tests); the levels at which the system is to be tested (i.e., at the system, subsystem, or component level); and the specific data required to evaluate the system, including test tolerances, data requirements, and analysis procedures. To ensure traceability to the design requirements, the System Test entity receives test planning guidance from the design capture views, since they are the repositories of design information against which the design must be verified. Test results are then used to annotate the design information in the capture views, so that a traceable record of test performance can be maintained as the design transitions to development.

4.1.4 Existing Design Capture

Existing Design Capture forms a description of the system's current design, including elements of each of the five capture views. The purpose of the existing design capture is to serve as a repository for information concerning the current status of an existing system's design. This entity is of value for designs which must be reengineered, as it provides access to the system design before modification. As a result, the design information can be imported directly into the five capture views without the need to retrace each view from its underlying requirements. With this existing design capture information, the reengineering design capture process can focus instead on capturing potential solutions to the revised or new requirements which specify the need for the reengineered system.

Since the key issue in reengineering is the ability to understand the functionality, behavior, and implementation of the existing system, the design capture views framework naturally supports this process. Depending on the level of reengineering, information about the existing system can be specified in the appropriate capture views. Supporting various levels of detailed descriptions as well as multi-levels of design abstraction, the capture views provide a framework for the whole spectrum of reengineering. For example, for the reuse of existing software, the functionality of the existing code can be captured in the software architecture of the Implementation Capture View. This software architecture provides a better understanding for the software function of the existing code which can then be assessed for its use in new required applications. For the

reengineering at system or subsystem level, the functionality, behavior, and implementation of the existing subsystem (or system) can be captured by the Functional, Behavioral, and Implementation Capture Views.

4.1.5 Development Engineering

Development Engineering is the process of transforming the system design from a detailed set of hardware, software and human operator specifications into a physical entity which contains all of the specified elements of the design, and which conforms to all system requirements and constraints. This entity thus has a strong relationship with the Environmental Capture View, in which the development, manufacture, maintenance, and logistics considerations reside. Thus, decisions made regarding configuration management strategies, program schedules and milestones, development staffing and facilities, and risk management approaches are all captured within the Development Engineering entity based on information from the Environmental Capture View. In return, the Development Engineering entity provides a description of the "as-built system", including any modifications made to accommodate the issues listed above, to the capture views for annotation of the design and traceability back to the Requirements Entity.

4.2 RELATIONSHIPS AMONG DESIGN CAPTURE VIEWS

This section describes the relationships among the system design capture views as defined to date. As can be seen from Figure 4-2, these relationships provide a means for establishing the kinds of information which flow between views. The following paragraphs describe in detail these inter-view relationships, as well as the relationships between each view and the external entities.

4.2.1 Informational Capture View

Figure 4-3 summarizes the inter-view relationships and external interfaces of the Informational Capture View.

4.2.1.1 Trace to Top Level System Behavior. At the top level, the system's desired behavior is used by the Informational Capture View to describe systems in conceptual terms to all stakeholders; as a result, the Informational Capture View's description of the system's behavior is traced to the top level conceptualization found in the Behavioral Capture View.
This is represented in the Informational Capture View through the use of a hierarchical Information Model which contains the relationships and required behavior of the system's component entities.

4.2.1.2 Trace to Functional View. The Functional Capture View is the store of the information describing the flow and transformation of data through the system. As such, an abstraction of this information (such as a conceptual description of an ASW system's tracking

FIGURE 4-2. TOP LEVEL DESIGN CAPTURE VIEWS INTERRELATIONSIPS

FIGURE 4-3. INFORMATIONAL CAPTURE VIEW RELATIONSHIPS

function) is used in the Informational Capture View to support the communication of functional design concepts to the entire systems engineering team, including the requiring organization, without constraint by specific implementation choices.

4.2.2 Functional Capture View

Figure 4-4 summarizes the inter-view relationships and external interfaces of the Functional Capture View.

FIGURE 4-4. FUNCTIONAL CAPTURE VIEW RELATIONSHIPS

4.2.2.1 Trace Functional Design to System Requirements. The system requirements are the source of the functional information described in Section 2.3. As a result, no information should be contained in the Functional Capture View which cannot be traced back to a statement of need or condition of constraint in the requirements set. For example, function or data flow descriptions should be clearly traceable to requirements found in high-level documents such as a Mission Needs Statement or Top-level Warfighting Requirements Document.

4.2.2.2 Provide Functional Design Description to System Design Evaluation. System Design Evaluation is the set of procedures and processes which the system design team and requiring organization perform to determine the degree of acceptability and improvement strategies for a candidate design. In order for the design to be evaluated, the Functional Capture

View must provide to the System Design Evaluation process a detailed functional design description. The Functional Capture View provides descriptions of the functions, data flows, and data stores associated with the candidate design. These are used in the System Design Evaluation process to determine the adequacy of the functional description *vis a vis* system requirements, and to verify (where applicable) that the system's functions can be performed within specified timing or resource utilization tolerances.

4.2.2.3 Allocate Functions to Implementation View. Described in detail in Section 2.4, this relationship is the process whereby the functions, data flows and data stores captured in the Functional Capture View are mapped to appropriate hardware, software, and human resources in the Implementation Capture View. The objective is to ensure full allocation of functions while providing an implementation which meets the system's physical goals (e.g., cost, weight, volume, complexity).

4.2.3 Behavioral Capture View

As described in Section 3.3, the Behavioral Capture View consists of three different levels of behavioral design information: top level, functional, and implementational. This section discusses the relationships of each of these in turn. Figure 4-5 summarizes the inter-view relationships and external interfaces of the Behavioral Capture View.

FIGURE 4-5. BEHAVIORAL CAPTURE VIEW RELATIONSHIPS

4.2.3.1 <u>Top Level Behavior</u>. As the top level system behavior is established, certain information can be traced. Following are the inter-view relationships and external interfaces of the top level system behavior.

4.2.3.1.1 TRACE SYSTEM DESIGN BEHAVIOR TO SYSTEM REQUIREMENTS. The behavior of the system as designed is traced back to the top level system requirements to provide a mechanism for verification that the design meets the top level behavioral requirements found in Mission Needs Statements and other high-level requirements documentation. For example, an automated air defense system may be required to launch a missile within 10 seconds of target detection. At the top level, this specific behavioral requirement is traced back to the Top Level Warfighting Requirement document which describes the overall system behavior and functionality.

4.2.3.1.2 PROVIDE SYSTEM BEHAVIOR DESIGN DESCRIPTION TO SYSTEM DESIGN EVALUATION. In order to conduct design optimization, simulation and analysis, the designed system's real time behavior is provided from the system-wide point of view, i.e., the system's top-level behavior is used as a means of quickly verifying if the candidate design meets its overall performance goals. Should the design not meet these goals, more detailed behavioral information is required from the functional and implementation behavior descriptions to locate, identify and help correct the shortfall.

4.2.3.1.3 PROVIDE THREAD DEFINITIONS TO FUNCTIONAL AND IMPLEMENTATION DESIGN BEHAVIOR. System Behavior Threads represent required real-time system behavior with an initiation point, a termination point and a measurable or identifiable outcome/result. The initiation point and termination point must be unambiguous identifiable events in space and time, such as a particular data flow or control flow. The outcome or result must be a measurable or identifiable consequence of the behavior triggered at the initiation point and completed at the termination point. Thus the system behavior threads combine a particular performance requirement or required system action with a timing requirement. The definitions of the required behavior threads derive from the top-level behavioral requirements expected of the system; hence, these definitions are provided to functional and implementation design behavior from the top level behavioral description.

4.2.3.2 <u>Functional Design Behavior</u>. The following paragraphs describe the inter-view relationships and external interfaces of the system functional behavior.

4.2.3.2.1 MAP TO IMPLEMENTATION DESIGN BEHAVIOR. The behavior of the functions described in the Functional Capture View must be included in the implementation definitions in the Implementation Capture View to fully capture the requirements associated with the candidate implementation. For example, a particular data storage unit (e.g., fixed disk drive) may perform the required function, but may not do so in the required time frame. For this reason, functional behavior is mapped to each hardware, software and human resource design decision made in the Implementation Capture View where such decisions have an impact on the system's behavior.

4.2.3.2.2 PROVIDE FUNCTIONAL BEHAVIOR DESIGN DESCRIPTION TO SYSTEM DESIGN EVALUATION. In the event that evaluation of top level behavior indicates that the system may not meet a behavioral requirement, the more detailed information regarding the functional behavior of the system must be made available to the System Design Evaluation process. In this way, a more thorough evaluation of the behavior of the system can be performed, supporting optimization or improvement of the behavioral threads which are causing the system to fall short of requirements.

4.2.3.2.3 SPECIFY FUNCTION REAL TIME BEHAVIOR TO FUNCTIONAL CAPTURE VIEW. As part of the functional descriptions contained in the Functional Capture View, the behavior of each function is captured in order to ensure that the function fulfills the necessary behavioral criteria.

4.2.3.3 Implementation Design Behavior. The following paragraphs describe the inter-view relationships and external interfaces of the system implementation behavior.

4.2.3.3.1 PROVIDE IMPLEMENTATION BEHAVIOR DESIGN DESCRIPTION TO SYSTEM DESIGN EVALUATION. In order to conduct design optimization, simulation and analysis, the designed system's implementation behavior is provided to System Design Evaluation, i.e., the behavior of a candidate implementation design is used to analyze the design's ability to meet the system's behavioral goals. Should the design not meet these goals, iterative optimization/improvement is performed, using the behavior threads defined in the top-level behavioral description as embodied in the implementation behavior design.

4.2.3.3.2 ALLOCATE DESIGN TO IMPLEMENTATION VIEW. As part of the implementation descriptions contained in the Implementation Capture View, the behavior of each hardware, software, and human resource is captured in order to ensure that the candidate implementation fulfills the necessary behavioral criteria.

4.2.4 Implementation Capture View

Figure 4-6 summarizes the inter-view relationships and external interfaces of the Implementation Capture View.

4.2.4.1 Trace Resource Constraints to System Requirements. The implementation of the system is traced back to the top level system constraints to provide a mechanism for verification that the design meets the constraints found in Mission Needs Statements and other high-level requirements documentation. For example, an ASW combat system may require the use of a variable-depth sonar system which can operate at depths up to 100 feet; this requirement places significant constraints on the design and fabrication of the hydrophone assembly, which must be watertight to at least this depth.

4.2.4.2 Trace Implementation Design Options to System Requirements. The Implementation Capture View must be able to store multiple candidate implementation designs to support evaluation. Each of these options arises from the top level requirements documentation,

FIGURE 4-6. IMPLEMENTATION CAPTURE VIEW RELATIONSHIPS

which describes the capabilities of the system in broad terms. For example, a requirements document for a tactical acoustic surveillance system may specify a need to sense a range of known threat frequencies; these frequencies, which may be detected by a range of hydrophone types, allow a variety of acceptable sensor implementations.

4.2.4.3 Provide Implementation Design Options to System Design Evaluation. The multiple implementation design options described above are compared against the system requirements in the system design evaluation process. These serve as a basis for simulation and analysis of the candidate design's capabilities and provide alternatives for situations in which the design under analysis fails to meet specified functional or behavioral requirements.

4.2.4.4 Provide Development Specification to Development Engineering. Once the system's implementation has been evaluated and optimized as necessary, design specifications must be prepared for use in the development of the full-scale system. These specifications serve as the baseline requirements for subsequent hardware and software design activities and form the basis for certification testing on the full-scale system.

4.2.5 Environmental Capture View

Figure 4-7 summarizes the inter-view relationships and external interfaces of the Environmental Capture View.

FIGURE 4-7. ENVIRONMENTAL CAPTURE VIEW RELATIONSHIPS

4.2.5.1 <u>Trace Environmental Description to System Requirements</u>. The system requirements are the source of the environmental information described in Section 2.3. As a result, no information should be contained in the Environmental Capture View which cannot be traced back to a statement of need or condition of constraint in the requirements set. For example, operational environmental descriptions, such as system concept-of-operations, should be clearly traceable to such high-level documents as a Mission Needs Statement or a Top-level Warfighting Requirements Document. Similarly, the support environment must be traceable to all development, manufacture, maintenance and logistics requirements, and must also conform with any required systems engineering processes or tools.

4.2.5.2 <u>Provide External Environmental Description to System Design Evaluation</u>. In order to conduct the activities associated with system design evaluation, detailed information is required regarding the operational environment. This includes information pertaining to the system's surroundings (i.e., geographic, meteorologic, and electromagnetic forces acting on the system) as well as the system concept-of-operations and all required external interfaces. Without this information, thorough analysis or optimization of the design would lack context.

4.2.5.3 <u>Provide Test Plan Guidance to System Test</u>. The Environmental Capture View is the repository for top-level test strategies which may impose constraints on the system design. This information describes the system-level tests which must be accomplished to ensure compliance with requirements before system development begins; the information consists of system performance metrics, critical testing issues, and the overall concept for system testing.

Since this information is crucial for conducting a test plan which verifies compliance, it must be available to activities conducted in the System Test entity to ensure that adequate system-level testing is performed.

4.2.5.4 Provide Development Guidance to Development Engineering. Decisions made regarding configuration management strategies, program schedules and milestones, development staffing and facilities, and risk management approaches are all captured within the Development Engineering entity. These decisions are based, in part, on an understanding of the development, manufacture, and logistics considerations in the Environmental Capture View. For example, the configuration management strategy for a nuclear reactor safety system (falling under the Nuclear Regulatory Commission's requirements for reactor electronics design and development) is very different from that used in a system developed and manufactured to best meet commercial standards.

4.2.5.5 Impact Functional Design in Functional Capture View. The Environmental Capture View captures information regarding several aspects of system design which affect the Functional Capture View. This is because the Environmental Capture View captures the operational environment of the system under design. Thus, the concept-of-operations bounds the top-level functionality of the system; the external systems and interfaces define functionality required to support linkage to the system's surroundings; and the environmental conditions constrain the functional design by requiring additional mitigating functionality (e.g., a combat system used in a nuclear electromagnetic pulse environment requires a pulse mitigation function).

4.2.5.6 Impact Behavioral Design in Top Level System Behavior. At the top level, the system behaves as a black box; the detailed behavior of individual functions or resources is not addressed. Instead, the system's behavior is described in terms of system level inputs and outputs. At this level, the system's desired behavior is captured by the operational environment as described in the Environmental Capture View; the concept-of-operations provides the philosophy and approach for the use of the system, including modes of operation and operational sequence diagrams; and the operational scenarios describe the conditions under which the system is expected to operate. Each of these components of the operational environment description provides information regarding the system's behavior at the conceptual level; e.g., an anti-submarine warfare (ASW) system may be required to engage multiple targets simultaneously. At this conceptual level, the system's desired behavior can be expressed in terms of time from detection to firing solution, or from detection to weapon launch. This top level behavioral description is defined in terms of operational environmental information captured in the Environmental Capture View.

4.2.5.7 Impact Implementation Design in Implementation Capture View. The specific resources used to implement the system design are influenced by the nature of the operational and support environment in which the system must eventually function. Specific information regarding each of these areas impacts the hardware, software and human resource choices made in the Implementation Capture View. For example, the environmental conditions may require that a communications system's ground terminals be designed for use regardless of weather conditions; this constrains the implementation by requiring ruggedization, dust exclusion, and components resistant to temperature extremes. Similarly, the external systems and interfaces may

include a connection to external systems which require a high frequency (HF) line of sight communications; again, this constrains the design by requiring that the data transmission path include an HF transceiver and antenna.

4.2.5.8 <u>Impact Informational Design in Informational View</u>. The Informational Capture View is the repository for conceptual descriptions of the system under design. The objective is to provide a means of communicating design concepts to all stakeholders in common terms. The Environmental Capture View provides the Informational Capture View with the necessary information to describe conceptually the operating and support environment such that the boundary and constraints of the design (conceptually) can be completely portrayed.

CHAPTER 5

DESIGN CAPTURE VIEWS AND EVALUATION TECHNIQUES

Each design capture view represents the system from a particular perspective and highlights different aspects of the design; however, they are not independent. The capture views represent different aspects of the same system and therefore, at a same level of abstraction, must be consistent. This does not prevent a particular capture view to be explored in more detail than another. Depending on the area of interest and/or depending on a certain stage of the design process, systems engineers can expand different capture views to search for alternative designs. The exploration results should also be merged back into a defined design baseline such that captured information can be maintained properly. The features of a particular capture view can directly or indirectly impact, to a greater or lesser degree, the design in another capture view, depending on how the relationship between the capture views is specified. Relationships defined between the design capture views, which were discussed in detail in Chapter 4, establish the nature of the inter-view dependencies. If very strong interdependencies and rules for linkage between the capture views are established, the system design is rapidly constrained when a single capture view is specified. If no formal linkage is established between the system capture views, then the capture views evolve independently which can result in severe inconsistencies.

The design capture views can be seen as a structured repository of design information. Regardless of the evaluation purposes or evaluation techniques, in order to assess the design, it is expected that evaluation models are constructed by extracting or deriving information from the design capture views. Detail descriptions of evaluation techniques and/or specific transition formats from design capture to design evaluation will not be addressed here. Instead, general concepts are discussed to show the usefulness of the capture views.

5.1 SYSTEM DESIGN EVALUATION DOMAIN

In order to construct analysis models, the captured information can be organized in various formats. The three evaluation domains in Figure 5-1 illustrate one way to gather the information. These evaluation domains are used to establish a basis for simulation and analysis of the system under design at the conceptual, logical, and implementation level.

FIGURE 5-1. DESIGN CAPTURE VIEWS AND ANALYSIS DOMAIN

5.1.1 Conceptual Domain

The Conceptual Domain includes the Environmental Capture View and the Informational Capture View. It highlights the system concept-of-operations from an operator's perspective using abstract object-oriented constructs. It also captures the expected operational environment, system requirements, and system design constraints. The Conceptual Domain provides a vehicle for the customer and the systems engineering team to form a clear mutual understanding of the proposed system operational requirements and constraints early in the design process and provides a mechanism for relating aspects of the design back to those requirements and constraints through all phases of the design process.

5.1.2 Logical Domain

The Logical Domain includes the Functional, the Behavioral, and the Environmental Capture View. It highlights the system design in terms of the functions, data, and control without regard to a specific set of physical resources. The Logical Model provides a vehicle for the systems engineering team to analyze the system design in terms of what the system must do functionally versus how it is accomplished in hardware and software. This partitioning allows the systems engineering team to deconvolve design issues and to address the logical (or non-implementation) aspects of the design.

5.1.3 Implementation Domain

The Implementation Domain includes the Implementation and the Environmental Capture View. It provides a vehicle for the systems engineering team to specify and capture the system design in terms of the specific hardware, software and human resources which perform the functions specified in the Logical Domain. It includes resource descriptions and a mapping which relates the functions, data flows and control from the Logical Domain directly to the resources captured in the Implementation Capture View. Timing information identified in the Behavior Capture View is also mapped to the resources of the Implementation Capture View providing a mechanism for examining critical system performance issues. Systems Engineers can explore many different mapping schemes (from functions and behaviors to resources) to search for the most suitable design.

5.2 SYSTEM DESIGN ANNOTATION

To enhance the evaluation results, the capture inforamtion must be described properly. Regardless of which design capture view is being constructed, systems engineers must have the capability to annotate the design information such that it can reflect goals and criteria of the requirements. An efficient technique to annotate the design is the System Design Factor (SDF). The study of the SDF has been conducted and documented by the Design Structuring and Resource Allocation Optimization Thrust.[20] Appendix A summarizes the SDF list that is pertinent to the system development process.

SDFs are attributes associated with any design element (DE) to strengthen its description. By definition, a DE is a set of one or more design components such as functions, data flows, states, objects, or relationships. Each SDF has one or more metrics defined to describes how the factor can be measured. Metrics can be derived from (1) past experience, (2) results of simulation/analysis/prototyping, (3) actual system measurements, (4) other SDF measurements, or (5) other SDF metrics. By attaching measurable values to the DE, the design can be evaluated more accurately. The classification of the metrics (i.e., simulated or measured) also supports systems engineers to understand the value of their analysis results. An example of a SDF template is illustrated in Figure 5-2.

5.3 DESIGN SIMULATION AND VIRTUAL SYSTEM PROTOTYPES

Design simulation provides a mechanism for analyzing system design trade-offs, determining the adequacy of a candidate design vis-a-vis requirements, and for addressing a broad spectrum of design concerns. If candidate designs are captured in a structured manner which supports rapid semi-automatic generation of design simulations, the need for time-consuming manual construction of design simulations may be avoided.

The central element of a Simulation Based Design approach to large, complex computer-based system development is the ability to quickly and efficiently simulate various

```
┌─────────────────────────────────────────────────────────────────────────────┐
│                                                                               │
│   1.   Name:      Reliability of Beam Former                                  │
│   2.   Type:      Probability                                                 │
│   3.   Range:     0.0 to 1.0                                                   │
│   4.   Units:     Units of Probability                                        │
│   5.   Methods/Principle:         Fault Tolerance, Highly Reliable Component  │
│   6.   Rationale:                 Life Critical Function                      │
│   7.   Relationship:              Availability, Fault Tolerance, ...          │
│        a. Relational Expression   Positive Correlation, Negative Correlation  │
│   8.   Quantification                                                         │
│        a. Type                                                                │
│        b. Formula          R(t) = 1 - F(t)        Actual                      │
│                            .989 entered           Required                    │
│                            1.01 * Required        Budgeted                    │
│   9.   Consistency Rule                                                       │
│        a. By aggregation Use Rule X and Y;                                    │
│           Rule X:  The probability of the component in series is the product of its │
│           probabilities.                                                      │
│           Rule Y:  The probability of the component in parallel use one of the rating, │
│           voting, scheme.                                                     │
│        b. By type                                                             │
│        c. By design factor                                                    │
│        d. By view                                                             │
│        e. By component                                                        │
│   10.  Reference:     Author's name                                           │
│   11.  Definition:    Text Book                                               │
│   12.  Annotation:    Comments                                                │
│   13.  Next Template:                                                          │
│                                                                               │
└─────────────────────────────────────────────────────────────────────────────┘
```

FIGURE 5-2. EXAMPLE OF SDF TEMPLATE

aspects of the design. This allows detailed analysis of specific portions of the design to isolate and resolve key design issues within the context of the overall system under design. Simulation of the design is accomplished through construction of virtual system prototypes and virtual environments. A virtual system prototype is defined as an executable computer model (e.g., analytic, Monte Carlo, etc.) which approximates the behavior, performance or other aspect of the system under design. A virtual environment is defined as an executable computer model which approximates the outside world (e.g., external interfaces, ambient environment, etc.) to the system under design for the purpose of stimulating a virtual system prototype.

The multi-domain design capture methodology summarized in the preceding chapters has been structured to support automatic rendering of virtual system prototypes and virtual environments based upon the design information captured within the five design capture views. These virtual system prototypes and virtual environments are then combined and executed (i.e., simulated) within the Simulation-Based Design simulation environment. Construction of virtual system prototypes is supported through employment of the five design capture views in the conceptual, logical, and implementation design domains.

5.3.1 Conceptual Simulation

Construction of a virtual system prototype within the conceptual design domain (hereafter referred to as the Conceptual Prototype) requires design information from the conceptual domain. The Conceptual Prototype models the system under design using objects (extracted from the Informational Capture View) which characterize key aspects of the system requirements and design constraints, and highlight the system concept of operations from an operational perspective. The Conceptual Prototype is simulated within one or more virtual environments created from information contained in the Environmental Capture View including external system interfaces, external objects, and the ambient environment. Simulation of the Conceptual Prototype within various virtual environments provides a vehicle for the customer and the systems engineering team to form a clear mutual understanding of the proposed system's operational requirements and constraints early in the design process and provides a mechanism for examining the adequacy and completeness of the system requirements within the expected operational scenarios.

A highly simplified description of a Conceptual Prototype in the conceptual domain is illustrated in Figure 5-3. A Conceptual Prototype of the combat system for a major surface combatant is constructed from objects including sensors, data fusion, communications, command and control, weapons, etc. Each of these objects contains attributes and methods which represent the required capabilities of the system under design as specified in the requirements and reflected in the Informational Capture View. For example, the sensor objects include data characterizing detection performance (i.e., array directivity, self noise, and target recognition differential) as well as methods for calculating sensor performance (i.e., the passive sonar equation). The virtual environment contains the ship on which the combat system is operating (including ship kinematic characteristics and associated systems), the acoustic environment of various ocean areas (including sound propagation conditions and environmental noise), friendly platforms, and threat platforms. Various engagements and operational scenarios are constructed to examine the warfighting contribution of the combat system as well as the operational effects and marginal utility of various trade-offs in specified system performance, dependability, etc. The Composite Area Analysis Model (CAAM)[21] and the Multi-Warfare Analysis and Research Systems (MARS)[22] are examples of the state-of-the-art in virtual environment representation. They provide a means of simulating these objects and environmental effects to support analysis of key trade-offs and approaches to optimizing system design factors.

5.3.2 Logical Simulation

Construction of a virtual system prototype within the logical design domain (hereafter referred to as the Logical Prototype) requires design information from the Logical Domain. The logical simulation allows systems engineers to analyze many key aspects of the design such as functional correctness, data flow bandwidths, safety, and security issues. Examples of tools that support this activity are Statemate,[23] Teamwork/Sim,[24] and CASCADE.[25]

RECAP: REQUIREMENTS CAPTURE & ANALYSIS

- DEFINE REQUIRED SYSTEM BEHAVIOR AND PERFORMANCE
- CHARACTERIZE CANDIDATE DESIGN OPTIONS
- ESTABLISH IMPLEMENTATION OPTION BOUNDARIES

CONCEPTUAL ANALYSIS

CONCEPT MODELING OUTPUTS
- WARFIGHTING CONTRIBUTION
- MARGINAL UTILITY
- SYSTEM LEVEL TRADE-OFFS

FIGURE 5-3. SIMULATION OF VIRTUAL SYSTEM PROTOTYPE IN THE CONCEPTUAL DOMAIN

Figure 5-4 illustrates a portion of a logical prototype constructed using the CASCADE modeling environment. An executable data flow and control flow representation of the system logical design is extracted from the Functional and Behavioral Capture Views and stimulated using an external environment and scenario description from the Environmental Capture View. In this example multiple subscribers on a military communications network are modeled. The message delay time and functional utilization were examined for cases with multiple subscribers. Simulation of this Logical Prototype provided a means for refining system functional partitioning and network control behavior.

5.3.3 Implementation Simulation

Construction of a virtual system prototype within the implementation design domain (hereafter referred to as the Implementation Prototype) requires design information from the Implementation Domain. The Implementation Prototype models the system under design in terms

FIGURE 5-4. SIMULATION OF VIRTUAL SYSTEM PROTOTYPES IN
THE LOGICAL DOMAIN

of its resources (hardware, software, and human operators). The Implementation Prototype
provides a simulation vehicle which can be used to estimate the system behavior for a given
implementation option. Figure 5-5 illustrates a portion of the hardware architecture for an
advanced submarine combat system captured within an automated software tool which supports
resource capture within the Implementation Capture View.[26] Each hardware resource is
represented by an object which contains attributes and methods characterizing the performance,
dependability and other aspects of the design. These attributes and methods are employed in
combination with the Logical Prototype to simulate the implementation design.

5.4 DESIGN OPTIMIZATION

Like simulation, optimization techniques can be applied at any system evaluation domain.
One should not assume that the optimization results provide an absolute optimal design; rather,
they guide systems engineers in narrowing the design space in such a way that it most satisfies the

FIGURE 5-5. SIMULATION OF VIRTUAL SYSTEM PROTOTYPES IN
THE IMPLEMENTATION DOMAIN

specified design requirements and criteria. The success of the optimization techniques relies on
the ability to specify the design information as well as the optimization objectives and criteria.

As summarized in Section 5.2, SDF plays a major role in supporting the specification for
optimization. It assists the engineer/customer in specifying the goals/criteria to be measured. Any
design element in the evaluation domain can be annotated using SDFs. A prototype,
DESTINATION,[27] was built to support the automation of the optimization process. The
following features are implemented in DESTINATION: (1) specification of the design goals and
parameters, (2) characterization of the evaluation domain (e.g., Logical or Implementation) of
interest, (3) specification of objective functions for optimization,[28] and (4) selection of
optimization methods.

Typical results of optimization include: recommendations for repartition of logical design,
suggestions for function-resource allocations, and suggestions for alternate implementation
architectures. These optimization results can then be verified by other evaluation techniques
including simulation and prototyping.

CHAPTER 6

AUTOMATION SUPPORT

Successful employment of any systems engineering methodology supporting a complex system design is largely a function of the degree of mechanization which can be achieved. The size and complexity of large-scale, advanced computer systems render manual application of most design processes or methods unusable. Considerable potential benefits can be gained from automation which supports a disciplined structured capture of the initial iteration of a system design and subsequent editing of that capture. Further significant efficiencies can be gained through automated consistency and completeness checking within and between the five system capture views which represent the captured design. These gains manifest themselves in three ways:

(1) **Productivity Gains** -- Designs are more quickly captured, analyzed and assessed.
(2) **Completeness** -- Designs are more completely described in a formalized manner.
(3) **Quality**-- Through increased analysis of alternative designs and abilities to quickly assess and optimize chosen designs through the automated support for design simulation and analysis within an integrated and highly automated design capture and analysis environment.

Previous chapters have addressed automation within design capture views. Therefore, in this section more emphasis will be placed on the integration between design activities including the ability to add/modify information from one design activity to another. Design activities may be particular design methods (e.g., structured analysis), design tools (e.g., CASE tool), design capture views (e.g., behavioral), or design processes (e.g., test). One critical aspect that will be' addressed is how information can effectively flow between the various parts that make up the design.

The issue of automation can be divided into a number of interrelated issues including the degree to which the process is automated, the flow of information in the automation, and the structure in which information is captured and controlled in an environment. The following sections discuss these issues and identify opportunities for technology development.

6.1 DEGREE OF AUTOMATION

It is critical for the integration of design activities that the proper mechanization of the methods is supported. This is especially true for large, complex, dependable, real-time, time-

critical systems. These large systems make it intractable to manually reenter information in multiple system design tools.

The degree of automation for an environment is a spectrum from totally manual annotation to complete automated annotation. Manual annotation transfers information between design activities via the system designer by manually determining and entering the proper information. The specification of the system design is separate from any of the specifications for system modeling, system validation, system test, etc. Updates and consistency are purely manual processes. In contrast, complete automation integration is performed without a designer's intervention.

In general, integration tends to become less time consuming and less prone to error as automation increases. Often as one aspect of the process is automated, an additional integration between design activities can be added to the environment. This new capability may be manual or automated. For instance, if the ability to move from a CASE oriented capture tool to a desktop publishing documentation tool is automated, then this may free time to (manually) update the CASE database as the documentation is modified in a desktop publishing package.

6.2 FLOW OF INFORMATION

For large, complex systems the ability to integrate the information captured in various methods is critical. The integration techniques employed between two design activities can be classified as supporting forward annotation, backward annotation, or full integration. In each successive mechanization listed above, the information between the activities is successively more unified.

6.2.1 Forward Annotation

Forward annotation is the process of transferring the pertinent information from one design activity to another. While forward annotation occurs to some extent during the design of all systems, one key to a successful design process is for efficient, robust forward annotation to be included in the process. Figure 6-1 pictorially shows the process. In an automated system design process, forward annotation often refers to the movement of information from a CASE tool to a system modeling tool.

Forward annotation is restrictive in the sense that the information flow is one way. If the information sent is changed, the total design specification becomes inconsistent. If the information is changed in the initial specification, then the forward annotation needs to be performed again. Also, by its nature, forward annotation allows multiple copies of the same information (possibly formatted or represented differently) to exist in various design activities. Because of this, forward annotation puts a significant strain on the configuration management activities of the system.

FIGURE 6-1. FORWARD AND BACKWARD ANNOTATION

Full forward annotation into a design activity occurs if all pertinent information for the activity can be derived from other activities. For example, if a CASE tool captures a complete system design, it could be forward annotated into a system modeling tool, providing all the necessary modeling specification information concerning the system.

6.2.2 Backward Annotation

As shown in Figure 6-1, backward annotation is a complementary process to forward annotation. Given information that has been forward annotated from Design Activity X to Design Activity Y and modified in Design Activity Y, then backward annotation is the process of updating the information in Design Activity X. Automated backward annotation does not currently exist to any large extent in today's development of systems. However, backward annotation is critically important in the development of complex systems, especially to maintain consistency between design activities.

6.2.3 Full Integration

Full integration has consistency of information maintained. The ultimate goal is to have this consistency concerning all factors and aspects of the system's specification. In the integration process, only the common information needed by the design activities are shared.

A forward annotation capability allows this information to be propagated to the appropriate design activities, when appropriate. If information currently defined by the systems engineer is not complete, the designer could add the information which would be forward and backward annotated as necessary. During the design process, information necessary for a type of systems modeling (not to be performed on the system) need not be specified.

6.3 STRUCTURES FOR REPRESENTATION

A number of strategies exist for structuring design information for integration. They fall into five main mechanisms:[29]

1. Value Added -- add addition functionality to tool;
2. Point to Point -- exchange via tool to tool interface;
3. Repository -- exchange via tool to central database;
4. Glue and Baling Wire -- exchange via combination of 1-3; and
5. Messaging -- exchange via domain specific set of messages.

All these mechanisms involve defining common data representations, translations between representations, and translations between tools. It is also important to define what information flows between tools automatically versus the flow between tools in a more controlled manner. In general, once the information is exchanged, each tool will have its own internal interfaces and data representation.

At this point it is not clear which approach is the best to provide high degrees of integration. Value Added and Point to Point solutions have been used for some time with limited success. Repositories are a more recent innovation, but have scalability, acceptance and consistency concerns. Glue and Baling Wire appears to be the current approach of many operational engineering environments where appropriate and feasible. Messages is a new approach which shows great potential, but has not been validated for diverse environments to date. The key issue for any of these environments is for domain knowledge information semantics to be clearly defined and formally described.

A number of ongoing efforts are trying to tie one or more of the approaches into an integration environment. Of most note are the ANSI X3H6 CASE tool integration models, Common Object Request Broker Architecture (CORBA), OSF Distributed Computing Environment (DCE), EIA Case Data Interchange Format (CDIF) and Portable Common Tools Environment (PCTE). Each of these efforts takes a slightly different approach to integration, and none provides complete integration by itself.

6.4 AUTOMATED SYSTEM DESIGN CAPTURE AND ANALYSIS TOOL DEVELOPMENT OPPORTUNITIES

Although some existing systems engineering tools are developed or enhanced (e.g., Teamwork, Statemate, RDD-100, etc.) to support the system development process, the state of the automation support for the multi-domain capture and analysis capability still needs to be improved. Following are the suggested opportunities for developing/enhancing the automated support to the described methodology.

6.4.1 Design Capture

An interactive graphic user interface with convenient pull-down menus and advanced editing functions such as **cut, paste, replicate** and **append** for complex design capture constructs is vital in capturing and modifying large, complex designs. The ability to manipulate entire sections of design element diagrams (including functional flow data diagrams, system behavior diagrams, and resource connection diagrams) should be provided. **Find** and **replace** is also a powerful feature which can significantly reduce the editing time required to address the inevitable design iterations of a large, complex design.

For the automation of the capture to be truly useful, it must be in a form that lends itself to analysis, but at the same time can be presented to the expected users in a manner that is understandable/natural. One important approach to help achieve this is the use of hierarchical design capture techniques which offer significant advantages in managing the complexity of large computer system designs through information hiding. A hierarchial structure allows the system capture views to be represented at various levels of detail, from a broad top level which encompasses the breadth and scope of the system and its external interfaces to very low levels which describe the details of a particular segment of the system design. Automated support for capturing the design capture views in a hierarchial manner and in navigating and accessing the various levels of the design represents a relatively low risk tool development effort.

Two additional critical issues for automation of design capture for large systems is the multiple user access and multiple versions/configuration control functions. As designs become large, the team involved in the development of the design also increases. To be able to work efficiently, team members must be able to capture the design in parallel. In addition, during design capture, alternative solutions are explored and designs are baselined. Tools must have the necessary mechanisms for this management of the design over time to be useful.

6.4.2 Intra- And Inter-View Consistency And Completeness Checking

Automatic consistency and completeness checking within each capture view can significantly reduce the effort required to maintain the design. Rules can be specified for checking the consistency and completeness within each design capture view. An example of the level of automation which is achievable in single domain consistency and completeness checking is the *Teamwork* CASE tool.[24] *Teamwork* provides automatic checking for DFDs which inform the user of unconnected data flows, functions without inputs and/or outputs, functions without process specifications, inconsistencies between levels of decomposition and data dictionary related problems. This type of automated internal view checking should be provided to the greatest extent possible within each of the five design capture views.

Consistency and completeness checking between the five capture views represents a large step toward a truly integrated design capture and analysis methodology. Rules specifying the consistency criteria between the various capture views, such as the ones proposed in the preceding section, would be checked in a manner similar to the intra-view checking described above.

Beyond the predefined checking criteria which can be built into a CASE tool, such as the *Teamwork* FFDD capabilities, is an ad hoc ability to check both individual design capture views and across design capture views. The ad hoc capability would allow a knowledgeable operator to specify certain rules for checking the design capture. This capability would provide the user significant flexibility in establishing the relationships between the capture views and would allow the user to establish liberal or strict relationships between the capture views.

6.4.3 Design Simulation

Automatic generation of executable simulations from the design capture and external environment descriptions contained in the five capture views represents potentially the largest efficiency gains which an effective methodology automation may provide. The cost of capturing and analyzing multiple design iterations and options is potentially prohibitive without the leverage provided by automation of the simulation and analysis process. Completed capture of the five capture views contains much of the information necessary to develop a simulation of both the system under design (i.e., the internal model) and the external environment (i.e, the external model) which is required to stimulate the system's internal simulation. Efficient use of the information captured in the five design capture views for automatic or semi-automatic generation of a system simulation represents a key capability in realizing the overall goal of efficient large-scale, complex system design development.

6.4.4 Document Generation

Automated generation of required system documentation from the captured design data also represents an area where existing design capture information can be efficiently employed to reduce the manual labor required to support the design development process. A variety of CASE tools currently provide this capability particularly in the software development arena. The automated document generation facility should provide standard formats (such as the MIL-STD-2167A DIDs) and also allow the user to specify the format and organization of the documents produced by the system.

6.4.5 Integrated Environments

Many of the activities in today's systems development are using manual annotation between design activities. For the most part, specifications of the system design (such as Teamwork, Software through Pictures (STP),[30] Statemate) and system modeling approaches (e.g., ADAS, SES Workbench, QASE) are entered separately. Some systems do allow limited access. For instance, Statemate has a built-in modeling system. Teamwork's SA (structured analysis) module with certain restrictions can be forward annotated into the ADAS simulation tools. The same is true for STP specification to SES workbench. However, typically multiple modeling approaches for differing pieces of the system design need to be performed in order to fully evaluate the design.

While many of the barriers to more complete integration are technical, there are also corporate, competitive reasons why the integration of methods has not occurred rapidly. Standards have been established to support the integration between tools. Among them, CDIF[31] and CORBA[32] are the two standards that are highly endorsed. Although these two standards have some impact in the tool developing industry, they are still not mature enough to support the identified needs.

REFERENCES

1.   DeMarco, Tom, *Structured Analysis and System Specification*, Prentice-Hall, Inc.,Yourdon Press, Englewood Cliffs, NJ, 1979.

2.   Hatley, Derek J. and Pirbhai, Imtiaz A., *Strategies for Real-Time System Specification*, Dorset House Publishing, New York, NY, 1987.

3.   Ward, Paul T. and Mellor, Stephen J., *Structured Development for Real-Time Systems*, Prentice-Hall, Inc., Yourdon Press, Englewood Cliffs, NJ, 1985.

4.   Shlaer, Sally and Mellor, Stephen J., *Object-Oriented Systems Analysis: Modeling the World in Data*, Prentice-Hall, Inc., Yourdon Press, Englewood Cliffs, NJ, 1988.

5.   *IEEE Standard VHDL Language Reference Manual*, IEEE Standard 1076-1987.

6.   Methodology and Tools Working Group, *ASW Architecture Development and Analysis Methodology, Ver. 1.0*, Technical Report, Honeywell Marine Systems Division, Everett, WA, Feb 1990.

7.   Molini, J. J, Maimon, S. K., and Watson, P. H., "Real-Time System Scenarios," *Real-Time Systems Symposium*, Florida, Dec 1990.

8.   Rumbaugh, James, et al., *Object-Oriented Modeling and Design*, Prentice Hall, Englewood Cliffs, NJ, 1991.

9.   *Software Requirements Specification for the System Engineer's Associate, A Computer-Aided Hierarchical Information Model Development and Analysis Tool*, Trident Systems Inc., Fairfax, VA, Aug 1991.

10.   Oliver, David, *Reengineering of Computer Based Systems*, The Third Annual Systems Engineering Technology Workshop, Naval Surface Warfare Center Dahlgren Division, Silver Spring, MD, Aug 1992.

11.   Harel, D., "Statecharts: A Visual Formalism for Complex Systems," *The Weizmann Institute of Science Technology Report*, Israel, Jul 1986, (also in *Science of Programming* 8, 1987).

12.   Jahanian, F., Lee, R. S., and Mok, A. K., "Semantics of Modechart in Real-Time Logic," Proc. 21st Hawaii International Conf. on Systems Sciences, Jan 1988.

REFERENCES (Cont.)

13.   Hoare, C. A. R., *Communicating Sequential Processes*, Prentice/Hall International, Englewood Cliffs, NJ, 1985.

14.   Spivey, J. M., *The Z Notation - A Reference Manual*, Second Edition, Prentice Hall International, UK, 1992.

15.   Britton, D. E., "CASCADE: a Dynamic Visual Environment for the Design of Computers," Master's Thesis, George Mason University, May 1990.

16.   Alford, M., "SREM at the Age of Eight," *IEEE Computer*, April 1985, pp 36-46.

17.   Reisig, W., "Petri Nets An Introduction," *EATCS Monographs on Theoretical Computer Science*, Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, 1982.

18.   Casey, M., Karangelen, N., and Bailey, S., *Resource Capture Prototype User's Manual*, Trident Systems Inc., Fairfax, VA, 1994.

19.   Karangelen, N. and Hoang, N., *Environmental Capture View Approach*, NSWCDD Interim Report, 1993.

20.   Nguyen, C., and Howell, S., *Systems Design Factors: The Essential Ingredients of System Design, Version 0.4*, NSWCDD/TR-92/268, Mar 1994, Naval Surface Warfare Center Dahlgren Division, Silver Spring, MD.

21.   Densford, D., *Composite Area Analysis Model Reference and User's Manual*, Trident Systems Inc., Fairfax, VA, 1994.

22.   Ng, H., and Wong, K., *The Multi Warfare Assessment and Research System (MARS) Requirement Specification, Version 1.5*, NSWCDD TR 91-746, Naval Surface Warfare Center Dahlgren Division, Dahlgren, VA, Dec 1991.

23.   *Statemate User Reference Manual*, i-Logix, Inc., Burlington, MA, 1992.

24.   *Teamwork/SA User's Guide, Rel. 4.0*, Technical Documentation Department of Cadre Technologies Inc., Cadre Technologies Inc., Providence, RI, Dec 1990.

25.   Britton, D. E., et al., *CASCADE Software Requirements Specification*, Trident Systems Inc., Fairfax, VA, 1993.

26.   Hoang, N. and Karangelen, N., *Implementation Capture View Approach*, NSWCDD Interim Report, 1993.

REFERENCES (Cont.)

27. Nguyen, C., Howell, S., and Hwang, P., "System Design Structuring and Allocation Optimization," *Proceedings of the 1991 Systems Design Synthesis and Technology Workshop*, Sept 1991, Silver Spring, MD, pp. 117-128

28. Mansour, N. and Fox, G., "Physical Optimization Methods for Allocating Data to Multicomputer Nodes," *Proceedings of the 1991 Systems Design Synthesis and Technology Workshop*, Sept 1991, Silver Spring, MD, pp. 169-182.

29. Epperson, R. E., "Integration Strategies and Technologies for Computer-Assisted System Engineering Environments," *Proceedings of the Fourth Annual International Symposium of the National Council on Systems Engineering*, Aug 10-12, 1994, San Jose, CA.

30. Interactive Development Environments, *Software Through Pictures Reference ManualSoftware, Ver. 4.2*, Interactive Development Environments, Inc., San Francisco, CA, 1988.

31. CASE Data Interchange Format Overview, CDIF Technical Committee, Electronic Industries Association, Jan 1994, ISBN 0-7908-0012-8.

32. The Common Object Request Broker: Architecture and Specification, Object Management Group and X/Open, ISBN 0-471-58792-3.

APPENDIX A

SYSTEM DESIGN FACTORS

1 PERFORMANCE
      1.1 RESPONSE TIME
      1.2 CAPABILITY
      1.3 RELATIVE ACTIVITY
      1.4 SPEED
      1.5 THROUGHPUT
      1.6 LATENCY
      1.7 LOAD BALANCING
            1.7.1 INFORMATION OVER LOAD
            1.7.2 PROCESSING OVER LOAD
      1.8 GRACEFUL DEGRADABILITY
      1.9 EFFICIENCY
      1.10 PREDICTABILITY

2 REAL-TIME
      2.1 HARDNESS
      2.2 HARD DEADLINES
            2.2.0.1 PERIODIC
            2.2.0.2 APERIODIC
            2.2.0.3 SPORADIC
      2.3 SOFT DEADLINES
            2.3.0.1 PERIODIC
            2.3.0.2 PERIODIC
            2.3.0.3 SPORADIC
      2.4 TEMPORAL DISTANCE
      2.5 TARDINESS
      2.6 NUMBER OF CONSECUTIVELY MISSED DEADLINES
      2.7 PREDICTABILITIES
      2.8 GRACEFUL DEGRADATION

3 COMPUTATION/PROCESSING REQUIREMENTS
      3.1 IMPORTANCE
      3.2 USEFULNESS
      3.3 PRIORITY
      3.4 (COMPUTING) PORTABILITY
      3.5 INTERRUPT/RESET CAPABILITIES

4 DEPENDABILITY
      4.1 RELIABILITY
      4.2 ACCURACY
      4.3 FAULT TOLERANCE
      4.4 GRACEFUL DEGRADABILITY
      4.5 REDUNDANCY

4.5.1 STATIC
4.5.2 DYNAMIC
4.6 AVAILABILITY
    4.6.1 INHERENT AVAILABILITY
    4.6.2 ACHIEVED AVAILABILITY
    4.6.3 OPERATIONAL AVAILABILITY
    4.6.4 EASE OF REPLACEMENT
    4.6.5 CRASH RECOVERABILITY
    4.6.6 COMPUTATION HEAVY PROCESS EFFECTS
4.7 QUALITY

5 SECURITY
5.1 CLASSIFICATION
5.2 TYPE OF DATA
    5.2.1 LEVEL I (CLASSIFIED)
        5.2.1.1 TOP SECRET OR ABOVE
        5.2.1.2 SECRET
        5.2.1.3 CONFIDENTIAL
    5.2.2 LEVEL II (SENSITIVE)
        5.2.2.1 PRIVACY ACT/FINANCIAL
        5.2.2.2 FOR OFFICIAL USE ONLY
        5.2.2.3 SENSITIVE MANAGEMENT
        5.2.2.4 PROPRIETY/PRIVILEGED
    5.2.3 LEVEL III (NONSENSITIVE)
        5.2.3.1 (OTHER--NOT CATEGORIES IN LEVEL I AND II)
5.3 PERCENTAGE OF PROCESSING TIME
5.4 ENCRYPTION TYPE REQUIREMENTS
5.5 IMPLEMENTATION TECHNIQUES REQUIREMENTS

6 HUMANWARE
6.1 EASE OF USE
6.2 POTENTIAL OPERATOR DECISIONS
6.3 1.OPERATOR DELAY / 2.USER RESPONSE TIME
6.4 OPERATOR ACTION(S)
6.5 1.REQUIRED NUMBER OF OPERATORS / 2.NUMBER OF SIMULTANEOUS USERS
6.6 USER INTENSITY
6.7 AVERAGE TIME FOR EACH CATEGORY
6.8 POTENTIAL ERRORS

7 PHYSICAL REQUIREMENTS
7.1 SIZE REQUIREMENTS
    7.1.1 HEIGHT
    7.1.2 WIDTH
    7.1.3 LENGTH
    7.1.4 DEPTH
    7.1.5 AREA
    7.1.6 VOLUME
7.2 WEIGHT REQUIREMENTS
7.3 RUGGABILITY (RUGGEDIZED)
7.4 SURVIVABILITY
7.5 (PHYSICAL) PORTABILITY
7.6 ENERGY REQUIREMENTS
    7.6.1 (ENERGY) CONSUMPTION
        7.6.1.1 ELECTRICAL (ENERGY CONSUMED)
        7.6.1.2 FUEL (ENERGY CONSUMED)
        7.6.1.3 OTHER (ENERGY CONSUMED)

APPENDIX B

PASSIVE SONAR SYSTEM EXAMPLE
VERSION 0.03

The intention of this example is to provide a vehicle for discussing the design capture methods. It is presently incomplete and contains some inconsistency errors. This should be considered a working document which will be subject to considerable revision.

CONTENTS

SECTION 1

TOP LEVEL SYSTEM REQUIREMENT

This description is an edited version of the Molini paper sample passive sonar system description with some embellishments. The intent is to add some detail to the existing text without creating a classified document.

1.0 General System Description

Passive sonar systems are used for determining the presence, location, or nature of objects in the sea from underwater sound the objects emit. Active sonar transmits an acoustic signal which, when reflected from a target, provides the sonar receiver with a signal. Based on this received signal, detections and position estimates are made. Passive sonar bases its detection and estimation on sounds that emanate from the target itself.

In passive sonar, and the receiving subsystem of active sonar, the received acoustic waveform from each hydrophone consists of one or more signals and background noise. The hydrophone converts the acoustic waveform to an electronic signal. The signals are amplified, filtered, sampled, and digitized in a signal conditioner. The digitized hydrophone outputs from the signal conditioner are combined by a digital beamformer to form a set of beams. A beam increases the sound from the beam direction and reduces the sound from other directions. Beam data are then processed to obtain detection and estimation statistics. Based on the values of these statistics, the system decides where targets are located. Detected targets are tracked by modifying the beamformer parameters and steering a beam toward the target.

A detected target is also analyzed. Analysis will include classification: distinguishing a signal returned from a target with regard to the type of target that produced the signal. A target is classified by the signal frequency spectrum and dynamics (for example, a school of fish versus a submarine) of its target signal.

The passive sonar system which is the subject of this example top level requirements specification is designed to detect and track submarine targets that can operate at any speed from 0 to 15 knots at any depth from 0 to 500 feet. The target sound source is assumed to consist of both low frequency broadband noise and vibrational harmonics from rotating machinery with a nominal rotation rate of 2,400 rpms. Therefore, it is expected that the signal from a submarine contains both broadband a fundamental 40-hertz component and the first three harmonics.

The data processing load for a sonar system is roughly proportional to the number of hydrophones, number of beams, and the sample rate. The larger the array of hydrophones the greater the beamforming gain and the greater the detection range. The finer the beam width, the finer the display resolution that assists in resolving multiple targets even though they appear close together. Some of the beams are used for detection displays, some for tracking targets, and some just for listening. Typically, the number of detection beams is fixed, so the detection processing varies little over time. The number of steered tracker beams may change as targets come and go. When the number of tracker beams is changed, the tracking load changes proportionally.

2.0 Example Passive Sonar System Performance Requirements

This section summarizes the top level performance requirements for the example sonar system in quantifiable terms. Each requirement represents a potential testing criteria for system acceptance test.

The sonar system shall:

1.   Operate concurrently to detect, track, localize, and classify surface and submarine targets.

2.   Achieve a broadband passive Figure of Merit (FOM) of TBD dB at a geometric mean frequency of TBD hz in the band from TBD hz to 256 hz against a TBD dB//1hz//1upas source level target.

3.   Achieve a narrowband passive Figure of Merit (FOM) of TBD dB TBD db//1hz//1upas source level target between the frequencies of TBD hz to 256 hz.

4.   Search continuously in a passive mode 360 degrees in azimuth and 120 degrees in D/E.

5.   Passively track up to 100 targets simultaneously.

6.   Analyze for classification purposes 12 targets simultaneously.

7.   Achieve a passive bearing error of less than TBD degrees at a signal SNR of TBD dB.

8.   Achieve an active bearing error of less than TBD degrees and a range error of TBD % of range.

9.   Hull mounted array must fit within a spherical area less than TBD ft in diameter.

10.   Towed array must be a retractable line array of less than TBD ft in length.

11.   Power requirements must be less than TBD amps at 120 volts, 60 hz.

12.   A minimum of TBD and a maximum of TBD operators shall operate the system in any operational condition.

13.   Achieve a MTBF of TBD for the full-up system. Full-up system is defined as all system functions available at full capacity.

14.   Achieve an MTBF of TBD for critical capabilities. Critical capabilities are defined as: (1) passive detection in 360 degrees in azimuth and 60 degrees in D/E and (2) concurrent tracking of up to 20 targets.

15.   Audio shall be provided for up to four headsets for operator selected beams and trackers. Audio shall be synchronized with displayed video data within 100 msec and shall slip no more than 100 msec over five hours to support high fidelity reconstruction.

16.   Passive trackers shall be updated at a one sec rate.

17.   Navigational position, attitude, heading and speed data shall have a latency of less than 200 msec throughout the system.

18.   Time synchronization shall reach all functions such that significant errors (> 2.5%) are not generated due to delays.

19.   Operator initiated automated detection and classification aids shall have a response time of less than one sec from operator request to displayed result.

20.   Passive detection display data shall be computed and displayed at least four times per sec.

21.   Display of detection data shall be synchronized to within 100 msec of the corresponding audio data.

22.   All operator controlled display cursor positions and associated read outs will be updated at a 10 hz rate.

SECTION 2

INFORMATIONAL CAPTURE VIEW

Hierarchical Information Model Notation

OBJECT'S NAME

Object which is internal to the system

OBJECT NAME

Object which is internal to the system contains additional detail on another diagram and is part of the title object.

OBJECT'S NAME

Object which is internal to the system and is part of the title object. (Title object appears in top left conner of the diagram)

OBJECT'S NAME

Object which is external to the system

OBJECT NAME

Object which is internal to the system and contains additional detail on another diagram

OBJECT NAME

Object which contains additional detail on another diagram and is external to the system

POWER SUPPLY — Supply Power — **BEAMFORMER**
1:1   1:3

A POWER SUPPLY Supplies Power to from 1 to 3 BEAMFORMER at a time

A BEAMFORMER is supplied power by only 1 POWER SUPPLY

The existence of the BEAMFORMER depends on the existence of the POWER SUPPLY

SIGNAL PROCESSOR — Detect — **TARGET**
1:5   0:N

A SIGNAL PROCESSOR can detect many target
A TARGET can be detected by 1 to 5 SIGNAL PROCESSOR

SUPERTYPE

SUBTYPE   **SUBTYPE**

Subtype objects inherit everything, without exception, from their supertype object. Additional attributes may be added to the subtype object. If so, subsequent subtypes inherit the additional attributes.

B-7

Hierarchical Information Model of the Passive Sonar Example

Context_Diagram: Passive Sonar System

POSITIONING SYSTEM

MASTER CLOCK

CIC

PASSIVE SONAR

TARGET

SUR

SUB

FALSE

Provide Time 1:1

1:N

Provide Location 1:1

1:N

Provide Tracking Information 1:N

1:N

Detect 1:N 0:N

Classify 1:N 0:N

Track 1:N 0:N

Object's Attribute List for the Passive Sonar Example

1. Acoustic Processor
2. Analog Signal Conditioning Equipments
3. Application Processor
4. Beamformer
5. CIC
6. Display Console
7. GEO Reference
8. Hull Penetrators & Cable
9. Hydrophone Array
10. Master Clock
11. Network
12. Operator
13. Passive Sonar
14. Positioning System
15. Power Supply
16. Signal Processor
17. System Support Service
18. Target
19. Target_False
20. Target_Sub
21. Target_Suf
22. Time Base

NAME: **Acoustic Processor**

ATTRIBUTE:
*   CPU Type:
*   Model Number:
        Bus Attachment:
        Memory Size:
        Power Supply Type:
        Size:
        Interface Type:

NAME: **Analog Signal Conditioning Equipments**

ATTRIBUTE:
*   Signal Conditioner Type:
*   Model Number:
        Number of Input Channel:
        Input Signal Dynamic Range:
        Output Signal Amplitude Range:
        Output Signal Bandwidth per Channel:
        Power Supply Type:
        Size:
        Input Interface Type:
        Output Interface Type:

NAME: **Application Processor**

ATTRIBUTE:
*   CPU Type:
*   Model Number:
        Bus Attachment:
        Memory Size:
        Power Supply Type:
        Size:
        Interface Type:

NAME: **Beamformer**

ATTRIBUTE:
*   Beamformer Type:
*   Model number:
        Number of Output Beam:
        Type of Output Beam:
        Number of Input Hydrophone:
        Power Supply Type:
        Size:
        Input Interface Type:
        Output Interface Type:

NAME: **CIC**

ATTRIBUTE:
*   ID number:
*   Combat System Type:
        Interface Bandwidth:
        Interface Type:

NAME: **Display Console**

ATTRIBUTE:
*   Display Console Type:
*   Model Number:
        Interface Type:
        Interface Bandwidth:

Compatible Computer Type:
Power Supply Type:
Size:

NAME: **GEO Reference**

ATTRIBUTE:
*   GEO Type:
*   ID number:
        Position Update Rate:
        Position Format:
        Interface Bandwidth:
        Interface Type:
        Accuracy:

NAME: **Hull Penetrators & Cables**

ATTRIBUTE:
*   Cable Assembly Type:
*   Cable Assembly Model:
        Cable Assembly Impedance:
        Cable Type:
        Connector Type:
        Size:

NAME: **Hydrophone Array**

ATTRIBUTE:
*   Hydrophone Array Type:
*   Model Number:
        Array Dimension:
        Frequency Range:
        Beam Pattern:
        Operating Depth:
        Number of Hydrophone:
        Hydrophone Response Resonance Frequency:
        Interface Type:
        Communication Type:
        Cable Connector Type:

NAME: **Master Clock**

ATTRIBUTE:
*   Master Clock Type:
*   Model Number:
        Model:
        Format:
        Accuracy:

NAME: **Network**

ATTRIBUTE:
*   Network Type:
        Model Number:
        Bandwidth:
        Interface Type:
        Network Protocol:
        Protocol:

NAME: **Operator**

ATTRIBUTE:
*   Operator Type:

Name:
Rank/Rate:
Qualification:
Organization:

NAME: **Passive Sonar**

ATTRIBUTE:
*   ID number:
         Passive Sonar Type:
         Passive Sonar Model:

NAME: **Positioning System**

ATTRIBUTE:
*   Positioning System Type:
*   Model Number:
         Position Format:
         Position Update Rate:
         Accuracy:

NAME: **Power Supply**

ATTRIBUTE:
*   Model Number:
*   Power Type:
         Voltage Level:
         Current Capacity:
         Format:

NAME: **Signal Processor**

ATTRIBUTE:
*   Model Number:
         Acoustic Processor Type:
         Application Processor Type:

NAME: **System Support Service**

ATTRIBUTE:
*   ID number:
         Power Supply Type:
         Time Base Type:
         GEO Reference Type:

NAME: **Target**

ATTRIBUTE:
*   Target Type:
         Displacement:
         Hull Number:
         Location:
         Course:
         Speed:

NAME: **Target_False**

ATTRIBUTE:
         Duration:

NAME: **Target_Sub**

ATTRIBUTE:
        Depth:
        Depth Rate:

NAME: **Target_Sur**

ATTRIBUTE:
        High:
        Draft:

NAME: **Time Base**

ATTRIBUTE:
*   Time Base Type:
*   ID Number:
        Time Format:
        Time Update Rate:
        Accuracy:
        Interface Type:

SECTION 3

FUNCTIONAL AND BEHAVIORAL CAPTURE VIEW

Data Flow Diagram of the Passive Sonar System

DFD Context_Diagram PASSIVE_SONAR_SYSTEM
DFD 0   PASSIVE_SONAR_SYSTEM_PERFORMANCE
DFD 2   SIGNAL_PROCESSING
DFD 2.4   SIGNAL_CONDITIONING
DFD 2.5   ANALYSIS_AND_CLASSIFICATION
DFD 2.5-s0   ANALYSIS ACTIVATION
DFD 2.5-s1   ANALYSIS STATE
DFD 4   DATA FORMATTING AND OPTION SELECTION

Context-Diagram;6
PASSIVE_SONAR_SYSTEM

0;5
PASSIVE_ SONAR_ SYSTEM_ PERFORMANCE

FIX

STABILIZATION

1

Position

Selected_Information

DATA
FORMATTING
AND
OPTION
SELECTION

4

CIC_Track

SIGNAL_
PROCESSING

2

Processed_Data

Analysis_Control

GMT

TIME_
SYNCHRONIZATION

3

Time

Sample_Synch

Electronic_Signal

Acoustic_Signal

ACOUSTIC
ENERGY
CONVERSION

5

2;6
SIGNAL_PROCESSING

2.4;3
SIGNAL_ CONDITIONING

2.5.2
ANALYSIS_ AND_ CLASSIFICATION

2.5-s0;1
ANALYSIS ACTIVATION

Activate_Signal	1	2	3
"Activate_Uncued_Analysis"	1	0	2
"Activate_Cued_Analysis"	0	1	2
	AUTO SELECT ANALYSIS BEAMS	*OPERATOR SELECT ANALYSIS BEAMS*	*COMPARE SIGNATURE*

**2.5-s1;2
ANALYSIS STATE**

**UNCUED
ANALYSIS**

1

Analysis_Control="FALSE"/
Activate_Signal=
"Activate_Uncued_Analysis"

Analysis_Control="TRUE"/
Activate_Signal=
"Activate_Cued_Analysis"

**CUED
ANALYSIS**

2

4;12
DATA FORMATTING AND OPTION SELECTION

Detect_Data → **DETECT DATA FORMATTING .1** → Detect_Info → **DETECT DECISION MAKING .2** ← Sound → Detect_Decisions → **DETECT SELECT DATA FORMATTING .3** → Detect_Selects / Audio_Selects

Track_Data → **TRACK DATA FORMATTING .4** → Track_Info → **TRACK DECISION MAKING .5** ← Sound → Track_Decisions → **TRACK SELECT DATA FORMATTING .6** → Track_Selects / Audio_Selects

Class_Data → **CLASS DATA FORMATTING .7** → Class_Info → **CLASS DECISION MAKING .8** ← Sound → Class_Decisions → **CLASS SELECT DATA FORMATTING .9** → Class_Selects / Audio_Selects

Analysis_Control

Process Specification of the Passive Sonar System

0   Passive Sonar System Performance
1   Stabilization
2   Signal Processing
                2.1 Beamforming
                2.2 Tracking
                2.3 Audio
                2.4 Signal Conditioning
                        2.4.1 Amplify
                        2.4.2 A/D Conversion
                        2.4.3 Filter
                        2.4.4 Automatic Gain Control
                2.5 Analysis and Classification
                        2.5.1 Auto Select Analysis Beam
                        2.5.2 Operator Select Analysis Beam
                        2.5.3 Compare Signature
                2.6 Detection
3   Time Synchronization
4   Data Formatting and Option Selection
                4.1 Detect Data Formatting
                4.2 Detect Decision Making
                4.3 Detect Select Data Formatting
                4.4 Track Data Formatting
                4.5 Track Decision Making
                4.6 Track Select Data Formatting
                4.7 Class Data Formatting
                4.8 Class Decision Making
                4.9 Class Select Data Formatting
5   Acoustic Energy Conversion

NAME AND TITLE: **0**   **Passive_Sonar_System_Performance**

CHILD PROCESSES:
> Stabilization
> Time_Synchronization
> Signal_Processing
> Data_Formatting_and_Option_Selection
> Acoustic_Energy_Conversion

INPUT:
> Fix
> GMT
> Acoustic_Signal

OUTPUT:
> CIC_Tracks

OUTPUT TRANSFORMATION:
> CIC_Tracks <— FIX + GMT + Acoustic Signal.

PROCESS DESCRIPTION:
> Passive sonar equipment is used for determining the presence, location, or nature of objects in the sea from underwater sound the objects emit. Active sonar transmits an acoustic signal which, when reflected from a target, provides the sonar receiver with a signal. Based on this received signal, detections and position estimates are made. Passive sonar bases its detection and estimation on sounds that emanate from the target itself.

DESIGN CHARACTERISTIC:
> Sample Sonar System Top Level Requirements
>
> The sonar system shall:
> 1. Operate concurrently to detect, track, localize, and classify surface and submarine targets.
> 2. Achieve a broadband passive Figure of Merit (FOM) of TBD dB at a geometric mean frequency of TBD hz in the band from TBD hz to 256 hz against a TBD dB//1hz//1upas source level target.
> 3. Achieve a narrowband passive Figure of Merit (FOM) of TBD dB TBD db//1hz//1upas source level target between the frequencies of TBD hz to 256 hz.
> 4. Search continuously in a passive mode 360 degrees in azimuth and 120 degrees in D/E.
> 5. Passively track up to 100 targets simultaneously.
> 6. Analyze for classification purposes 12 targets simultaneously.
> 7. Achieve a passive bearing error of less than TBD degrees at a signal SNR of TBD dB.
> 8. Achieve an active bearing error of less than TBD degrees and a range error of TBD % of range.
> 9. Hull mounted array must fit within a spherical area less than TBD ft in diameter.
> 10. Towed array must be a retractable line array of less than TBD ft in length.
> 11. Power requirements must be less than TBD amps at 120 volts, 60 hz.
> 12. A minimum of TBD and a maximum of TBD operators shall operate the system in any operational condition.
> 13. Achieve a MTBF of TBD for the entire system.
> 14. Achieve an MTBF of TBD for critical capabilities (defined as passive detection and track). (all functions)
> 15. Audio shall be provided for up to four headsets for operator selected beams and trackers. Audio shall be synchronized with displayed video data within 100 msec and shall slip no more than 100 msec over five hours to support high fidelity reconstruction. (function 2 - Signal Processing)
> 16. Passive trackers shall be updated at a one sec rate. (function 2 - Signal Processing)
> 17. Navigational position, attitude, heading and speed data shall have a latency of less than 200 msec throughout the system. (function 1 Stabilization)
> 18. Time synchronization shall reach all functions such that significant errors are not generated due to delays. (function 3 - Time Synchronization)
> 19. Operator initiated automated detection and classification aids shall have a response time of less than one sec from operator request to displayed result. (functions 2 &3 - Signal Processing & Data Formatting and Option Selection)
> 20. Passive detection display data shall be computed and displayed at least four times per sec. (function 2 - Signal Processing).
> 21. Display of detection data shall be synchronized to within 100 msec of the corresponding audio data. (function 2 - Signal Processing)

DESIGN FACTOR:
    1. Performance
        1.1 Response Time: N/A
        1.2 Capability: TBD
        1.3 Relative Activity: N/A
        1.4 Speed: N/A
        1.5 Throughput: TBD
        1.6 Latency: N/A
        1.7 Efficiency: N/A
        1.8 Predictability: N/A

    2. Real-Time
        2.1 Deadlines
            2.1.1 Hard Deadlines: TBD
            2.1.2 Soft Deadline: TBD
        2.2 Synchronization:

    3. Computation/Processing Requirements
        3.1 Importance: N/A
        3.2 Usefulness: N/A
        3.3 Priority: N/A
        3.4 (Computing) Portability: N/A
        3.5 Interrupt/Reset Capabilities: N/A
        3.6 Memory Space: TBD

    4. Dependability
        4.1 Reliability: N/A
        4.2 Accuracy: N/A
        4.3 Fault Tolerance: N/A
        4.4 Availability
            Availability: N/A
            Inherent Availability: N/A
            Achieved Availability: N/A
            Operational Availability: N/A
            Ease of Replacement: N/A
            Crash Recoverability: N/A
            Computation Heavy Process Effects: N/A
        4.5 Quality: N/A

    5. Security
        5.1 Level: Unclass

DESIGN RATIONAL:

COMPATIBLE RESOURCE:

NAME AND TITLE: 1   **Stabilization**

CHILD PROCESSES:
    None

INPUT :
    Fix

OUTPUT:
    Position   1/sec

OUTPUT TRANSFORMATION:
    Position <— FIX.

PROCESS DESCRIPTION:
> Stabilization creates a position message which includes the instantaneous position and attitude vectors describing ship and sensor array latitude, longitude, heading/heading rate, roll/roll rate, pitch/pitch rate, speed, and depth/depth rate. These parameters are calculated based upon inputs from the ship's navigation system.

DESIGN CHARACTERISTIC:
> Position vectors shall reach the functions within 200 msec of the fix being taken by the ships navigation system.

DESIGN FACTORS:
> 1. Performance
>> 1.1 Response Time: 200 msec
>> 1.2 Capability: 10 (KIPS)
>> 1.3 Relative Activity: N/A
>> 1.4 Speed: N/A
>> 1.5 Throughput: N/A
>> 1.6 Latency: N/A
>> 1.7 Efficiency: N/A
>> 1.8 Predictability: N/A
>
> 2. Real-Time
>> 2.1 Deadlines
>>> 2.1.1 Hard Deadlines: n/a
>>> 2.1.2 Soft Deadline: 200 msec from fix taken to position vector available at user functions.
>> 2.2 Synchronization: N/A
>
> 3. Computation/Processing Requirements
>> 3.1 Importance: N/A
>> 3.2 Usefulness: N/A
>> 3.3 Priority: 3
>> 3.4 (Computing) Portability: N/A
>> 3.5 Interrupt/Reset Capabilities: N/A
>> 3.6 Memory Space:
>>> Input buffer (100 bytes + Output buffer (100 bytes+ Program storage (2 Kbytes)
>
> 4. Dependability
>> 4.1 Reliability: N/A
>> 4.2 Accuracy: N/A
>> 4.3 Fault Tolerance: N/A
>> 4.4 Availability
>>> Availability: N/A
>>> Inherent Availability: N/A
>>> Achieved Availability: N/A
>>> Operational Availability: N/A
>>> Ease of Replacement: N/A
>>> Crash Recoverability: N/A
>>> Computation Heavy Process Effects: N/A
>> 4.5 Quality: N/A
>
> 5. Security
>> 5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
> General Purpose Processor

NAME AND TITLE: 2 .   **Signal_Processing**

CHILD PROCESSES:

Beamforming
Tracking
Audio
Signal_Conditioning
Analysis_and_Classification
Detection

INPUT:

Position
Time
Sample_Synch
Selected_Information
Analysis_Control
Electronic_Signal

OUTPUT:

CIC_Track
Processed_Data

OUTPUT TRANSFORMATION:

CIC_Track <— Position + Time + Sample_Synch + Selected_Information + Electronic_Signal + Analysis_Control.
Processed_Data <— Selected_Information + Analysis_Control.

PROCESS DESCRIPTION:

Signal Processing provides for the conditioning, beamforming and analysis of multiple signal inputs from the systems acoustic sensors.

DESIGN CHARACTERISTIC:
TBD

DESIGN FACTORS:
1. Performance
    1.1 Response Time: N/A
    1.2 Capability: TBD
    1.3 Relative Activity: N/A
    1.4 Speed: N/A
    1.5 Throughput: TBD
    1.6 Latency: N/A
    1.7 Efficiency: N/A
    1.8 Predictability: N/A

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: TBD
        2.1.2 Soft Deadline: TBD
    2.2 Synchronization: TBD

3. Computation/Processing Requirements
    3.1 Importance: N/A
    3.2 Usefulness: N/A
    3.3 Priority: N/A
    3.4 (Computing) Portability: N/A
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space: TBD

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A

Inherent Availability: N/A
Achieved Availability: N/A
Operational Availability: N/A
Ease of Replacement: N/A
Crash Recoverability: N/A
Computation Heavy Process Effects: N/A
4.5 Quality: N/A

5. Security
5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
TBD

NAME AND TITLE: **2.1**   **Beamforming**

CHILD PROCESSES:
None

INPUT:
Position
Time
Samples: data_in
Class_Beams_Request
Audio_Beams_Request
Steering_data

OUTPUT:
Audio_Beams
Class_Beams
Detect_Beams
Track_Beams

OUTPUT TRANSFORMATION:
Audio_Beams <— Position + Time + Samples + Audio_Beams_Request.
Class_Beams <— Position + Time + Samples + Class_Beams_Request.
Detect_Beams <— Position + Time + Samples.
Track_Beams <— Position + Time + Samples + Steering_Data.

PROCESS DESCRIPTION:
Samples from the signal conditioning function for each hydrophone are delayed, scaled, and summed to form 120 fixed spatial beams for detection function processing and 100 steerable beams for track processing. All hydrophones are used to compute each beam.

DESIGN CHARACTERISTIC:
- Detection beam data (Detect_Beams) shall be computed and sent to the detection function at least 4 times per second.
- Track beam data (Track_Beams) shall be computed and sent to the track function at least 1 time per second.

- Class beam data (Class_Beams) shall be computed and sent to the Analysis and Classification function at least 1 time per second.

DESIGN FACTOR:
1. Performance
1.1 Response Time: 1 sec
1.2 Capability: TBD
1.3 Relative Activity: N/A

1.4 Speed: N/A
1.5 Throughput: N/A
1.6 Latency: N/A
1.7 Efficiency: N/A
1.8 Predictability: N/A

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: must keep up with the data provided by signal conditioning (i.e. A/D conversion rate of # of hydrophones * 32 bits * 512 samples per sec
        2.1.2 Soft Deadlines: detect beams and track beams provided at 250 msec and 1 sec rates respectively
    2.2 Synchronization: N/A

3. Computation/Processing Requirements
    3.1 Importance: N/A
    3.2 Usefulness: N/A
    3.3 Priority: 1
    3.4 (Computing) Portability: N/A
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space:
        Input buffer + Output buffer + Program storage

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A
        Ease of Replacement: N/A
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
    4.5 Quality: N/A

5. Security
    5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPES:

    Special Purpose Beamformer

NAME AND TITLE: **2.2**   **Tracking**

CHILD PROCESSES:
    None

INPUT:
    Position
    Time
    Track_Beams
    Track_Selects

OUTPUT:
    CIC_Tracks
    Steering_Data

Track_Data

OUTPUT TRANSFORMATION:
    CIC_Tracks <— Position + Time + Track_Selects + Track_Beams.
    Steering_Data <— Position + Time + Track_Selects + Track_Beams.
    Track_Data <— Position + Time + Track_Selects + Track_Beams.

PROCESS DESCRIPTION:
    - Track beam data provided from the beamformer is processed to determine the estimated bearing for each assigner tracker and to calculate beam steering coefficients to recenter each track beam on the estimated target position. The bearing rate for each tracker is also computed by a least squares fit of the bearing data over the most recent thirty second interval.
    - Trackers are initiated on bearings identified by the Track_Selects parameters.

DESIGN CHARACTERISTIC:
    - Up to 100 track beams shall be concurrently supported by the track function.
    - Track data (i.e. bearing, bearing rate and beam steering data shall be computed at least one time per second.
    - Less than one second shall elapse from the time a hydrophone sample is taken at the signal conditioning function to when the beam steering data is used by the beamformer to form the subsequent track beam.

DESIGN FACTORS:
    1. Performance
        1.1 Response Time: 1 sec
        1.2 Capability: 10 * number of track beam (KIPS)
        1.3 Relative Activity: N/A
        1.4 Speed: N/A
        1.5 Throughput:N/A
        1.6 Latency: N/A
        1.7 Efficiency: N/A
        1.8 Predictability: N/A

    2. Real-Time
        2.1 Deadlines
            2.1.1 Hard Deadlines: n/a
            2.1.2 Soft Deadline: Less than one second shall elapse from the time a hydrophone sample is taken at the signal conditioning function to when the beam steering data is used by the beamformer to form the subsequent track beam.
        2.2 Synchronization: N/A

    3. Computation/Processing Requirements
        3.1 Importance: N/A
        3.2 Usefulness: N/A
        3.3 Priority: 1
        3.4 (Computing) Portability: N/A
        3.5 Interrupt/Reset Capabilities: N/A
        3.6 Memory Space:
            Input buffer + Output buffer + Program storage

    4. Dependability
        4.1 Reliability: N/A
        4.2 Accuracy: N/A
        4.3 Fault Tolerance: N/A
        4.4 Availability
            Availability: N/A
            Inherent Availability: N/A
            Achieved Availability: N/A
            Operational Availability: N/A
            Ease of Replacement: N/A
            Crash Recoverability: N/A
            Computation Heavy Process Effects: N/A
        4.5 Quality: N/A

5. Security
    5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    General Purpose Processor

NAME AND TITLE: **2.3**   **Audio**

CHILD PROCESSES:
    None

INPUT:
    Position
    Time
    Audio_Beams
    Audio_Selects

OUTPUT:
    Audio_Beams_Request
    Sound

OUTPUT TRANSFORMATION:
    Audio_Beams_Request <--- Position + Time + Audio_Beams + Audio_Selects.
    Sound <--- Position + Time + Audio_Beams + Audio_Selects.

PROCESS DESCRIPTION:
    - Operator selected beams (track beams or detection beams) are converted to analog signals by reconstruction and filtering of the digital beam data.
    - Detection and track beams to be converted to analog audio signals are identified by the Audio cursor parameters.

DESIGN CHARACTERISTIC:
    - Any of the detection or track beams may be selected for audio reconstruction.
    - Less than one second shall elapse from the time a beam is identified by the audio cursor to when the beam data is converted to an analog audio signal.
    - Display of detection or track beam data shall be synchronized to within 100 msec of the corresponding audio data.

DESIGN FACTORS:
    1. Performance
        1.1 Response Time: 1 sec
        1.2 Capability: 10 * number of track beam (KIPS)
        1.3 Relative Activity: N/A
        1.4 Speed: N/A
        1.5 Throughput:
            - Input rate:
                - # Audio channels * 32 bits * 512 samples per sec
                - (position) lat 10 bytes
                                  long 10 bytes
                                  heading/heading rate 10 bytes
                                  roll/roll rate 10 bytes
                                  pitch/pitch rate 10 bytes
                                  speed 4 bytes
                                  depth rate 4 bytes
                                            * 4/sec
                - (time) 8 bytes * 4/sec
            - Output rate:
                Analog audio signals
        1.6 Latency: N/A

1.7 Efficiency: N/A
1.8 Predictability: N/A

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: n/a
        2.1.2 Soft Deadline:Less than one second shall elapse from the time a beam is identified by the audio cursor to when the beam data is converted to an analog audio signal.
    2.2 Synchronization: Display of detection or track beam data shall be synchronized to within 100 msec of the corresponding audio data.

3. Computation/Processing Requirements
    3.1 Importance: N/A
    3.2 Usefulness: N/A
    3.3 Priority: 3
    3.4 (Computing) Portability: N/A
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space:
        Input buffer + Program storage

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A
        Ease of Replacement: N/A
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
    4.5 Quality: N/A

5. Security
    5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    General Purpose Processor with digital to analog conversion capabilities

NAME AND TITLE: **2.4   Signal Conditioning**

CHILD PROCESSES:
    Amplify
    Filter
    Automatic Gain Control
    A/D Conversion

INPUT:
    Electronic_Signal
    Gain_Control_Signal
    Sample_Synch

OUTPUT:
    Samples

OUTPUT TRANSFORMATION:
> Samples <— Electronic_Signal + Sample_Synch + Gain_Control_Signal.

PROCESS DESCRIPTION:
> The signal conditioning function amplifies, filters and converts the hydrophone signals into synchronous discrete time samples and is the beginning of the acoustic signal processing pipeline.

DESIGN CHARACTERISTIC:
> - An independent channel of amplification filtering and A/D conversion is required for each hydrophone in the system..
> - The signal conditioning and beamforming functions must be synchronized within .0005 sec to support a 512 sample per sec rate.

DESIGN FACTORS:
> 1. Performance
>> 1.1 Response Time: 512 32 bit samples per sec per hydrophone
>> 1.2 Capability: TBD
>> 1.3 Relative Activity: N/A
>> 1.4 Speed: N/A
>> 1.5 Throughput: N/A
>> 1.6 Latency: N/A
>> 1.7 Efficiency: N/A
>> 1.8 Predictability: N/A
>
> 2. Real-Time
>> 2.1 Deadlines
>>> 2.1.1 Hard Deadlines: N/A
>>> 2.1.2 Soft Deadline:N/A
>> 2.2 Synchronization: The signal conditioning and beamforming functions must be synchronized within .0005 sec to support a 512 sample per sec rate.
>
> 3. Computation/Processing Requirements
>> 3.1 Importance: N/A
>> 3.2 Usefulness: N/A
>> 3.3 Priority:1
>> 3.4 (Computing) Portability: N/A
>> 3.5 Interrupt/Reset Capabilities: N/A
>> 3.6 Memory Space:
>>> TBD
>
> 4. Dependability
>> 4.1 Reliability: N/A
>> 4.2 Accuracy: N/A
>> 4.3 Fault Tolerance: N/A
>> 4.4 Availability
>>> Availability: N/A
>>> Inherent Availability: N/A
>>> Achieved Availability: N/A
>>> Operational Availability: N/A
>>> Ease of Replacement: N/A
>>> Crash Recoverability: N/A
>>> Computation Heavy Process Effects: N/A
>> 4.5 Quality: N/A
>
> 5. Security
>> 5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
> Special Purpose Processor

NAME AND TITLE: **2.4.1** **Amplifying**

CHILD PROCESSES:
None

INPUT:
Electronic_Signal
Sample_Synch

OUTPUT:
Amplified_Signal

OUTPUT TRANSFORMATION:
Amplified_Signal <--- Electronic_Signal + Sample_Synch.

PROCESS DESCRIPTION:
This function provides amplification of analog signals from each of the hydrophone channels.

DESIGN CHARACTERISTIC:
- Must handle 1600 hydrophone channels
- The frequency response of the amplifiers must be flat across the range from 10 hz to 250 hz and roll of at no more than TBD dB and must be matched such that the gain from hydrophone to hydrophone does not vary more than TBD dB.

DESIGN FACTORS:
1. Performance
    1.1 Response Time: 512 32 bit samples per sec per hydrophone
    1.2 Capability: TBD
    1.3 Relative Activity: N/A
    1.4 Speed: N/A
    1.5 Throughput: N/A
    1.6 Latency: N/A
    1.7 Efficiency: N/A
    1.8 Predictability: N/A

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: N/A
        2.1.2 Soft Deadline:N/A
    2.2 Synchronization: The signal conditioning and beamforming functions must be synchronized within .0005 sec to support a 512 sample per sec rate.

3. Computation/Processing Requirements
    3.1 Importance: N/A
    3.2 Usefulness: N/A
    3.3 Priority:1
    3.4 (Computing) Portability: N/A
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space:
        TBD

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A

Ease of Replacement: N/A
Crash Recoverability: N/A
Computation Heavy Process Effects: N/A
4.5 Quality: N/A

5. Security
5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
Special Purpose Processor

NAME AND TITLE: **2.4.2   A/D_Conversion**

CHILD PROCESSES:
None

INPUT:
Gain_Controlled_Signal

OUTPUT:
Samples

OUTPUT TRANSFORMATION:
Samples <--- Gain_Controlled_Signal.

PROCESS DESCRIPTION:
This function provides analog to digital conversion for each of the hydrophone channels.

DESIGN CHARACTERISTIC:
- The 1600 hydrophone channels will be converted concurrently to twelve bit digital samples at a 500 hz rate.

DESIGN FACTORS:
1. Performance
1.1 Response Time: 512 32 bit samples per sec per hydrophone
1.2 Capability: TBD
1.3 Relative Activity: N/A
1.4 Speed: N/A
1.5 Throughput: N/A
1.6 Latency: N/A
1.7 Efficiency: N/A
1.8 Predictability: N/A

2. Real-Time
2.1 Deadlines
2.1.1 Hard Deadlines: N/A
2.1.2 Soft Deadline:N/A
2.2 Synchronization: The signal conditioning and beamforming functions must be synchronized within .0005 sec to support a 512 sample per sec rate.

3. Computation/Processing Requirements
3.1 Importance: N/A
3.2 Usefulness: N/A
3.3 Priority:1
3.4 (Computing) Portability: N/A
3.5 Interrupt/Reset Capabilities: N/A
3.6 Memory Space:
TBD

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A
        Ease of Replacement: N/A
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
    4.5 Quality: N/A

5. Security
    5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    Special Purpose Processor

NAME AND TITLE: **2.4.3**   **Filter**

CHILD PROCESSES:
    None

INPUT:
    Amplified_Signal

OUTPUT:
    Filtered_Signal

OUTPUT TRANSFORMATION:
    Filtered_Signal <— Amplified_Signal.

PROCESS DESCRIPTION:
    This function provides band pass filtering for each of the hydrophone channels.

DESIGN CHARACTERISTIC:
    - The 1600 hydrophone channels will be band pass filtered concurrently across a frequency band from 10 hz to 250 hz with roll off of at least TBD dB per octave at the ends of the band.

DESIGN FACTORS:
1. Performance
    1.1 Response Time: 512 32 bit samples per sec per hydrophone
    1.2 Capability: TBD
    1.3 Relative Activity: N/A
    1.4 Speed: N/A
    1.5 Throughput: N/A
    1.6 Latency: N/A
    1.7 Efficiency: N/A
    1.8 Predictability: N/A

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: N/A
        2.1.2 Soft Deadline:N/A
    2.2 Synchronization: The signal conditioning and beamforming functions must be synchronized within .0005 sec to

support a 512 sample per sec rate.

3. Computation/Processing Requirements
    3.1 Importance: N/A
    3.2 Usefulness: N/A
    3.3 Priority:1
    3.4 (Computing) Portability: N/A
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space:
        TBD

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A
        Ease of Replacement: N/A
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
    4.5 Quality: N/A

5. Security
    5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    Special Purpose Processor

NAME AND TITLE: **2.4.4**   **Automatic_Gain_Control**

CHILD PROCESSES:
    None

INPUT:
    Filtered_Signal
    Gain_Control_Signal

OUTPUT:
    Gain_Controlled_Signal

OUTPUT TRANSFORMATION:
    Gain_Controlled_Signal <— Filtered_Signal + Gain_Control_Signal.

PROCESS DESCRIPTION:
    This function provides gain control signals to the hydrophone channel amplifiers which adjust the amplifier outputs to maintain an average level of voltage at the input of the A/D converters.

DESIGN CHARACTERISTIC:
    - Average level of voltage: TBD

DESIGN FACTORS:
    1. Performance
        1.1 Response Time: 512 32 bit samples per sec per hydrophone
        1.2 Capability: TBD

1.3 Relative Activity: N/A
1.4 Speed: N/A
1.5 Throughput: N/A
1.6 Latency: N/A
1.7 Efficiency: N/A
1.8 Predictability: N/A

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: N/A
        2.1.2 Soft Deadline:N/A
    2.2 Synchronization: The signal conditioning and beamforming functions must be synchronized within .0005 sec to support a 512 sample per sec rate.

3. Computation/Processing Requirements
    3.1 Importance: N/A
    3.2 Usefulness: N/A
    3.3 Priority:1
    3.4 (Computing) Portability: N/A
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space:
        TBD

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A
        Ease of Replacement: N/A
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
    4.5 Quality: N/A

5. Security
    5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    Special Purpose Processor

NAME AND TITLE: **2.5**   **Analysis_and_Classification**

CHILD PROCESSES:
    Operator_Select_Analysis_Beams
    Auto_Select_Analysis_Beams
    Compare_Signature

INPUT:

Position	4/sec
Time	4/sec
Class_Beams	1/sec
Class_Selects	aperiodic (2/ min)
Analysis_Control	aperiodic (2/ min)

OUTPUT:

    Class_Data
    Class_Beams_Request

OUTPUT TRANSFORMATION:

    Class_Beams_Request <— Position + Time + Class_Beams + Class_Selects + Analysis_Control.
    Class_Data <— Position + Time + Class_Beams + Class_Selects + Analysis_Control.

PROCESS DESCRIPTION:

    - Class beam data provided from the beamformer is processed to determine the characteristics of the acoustic energy in the selected beam and then compares those characteristics to known target signatures.
    - Analysis is initiated on track beams identified from two sources: (1) by the operator via the Class_Selects data flow and (2) automatically by the Analysis_and_Classification.

DESIGN CHARACTERISTIC:

    - Up to 12 class beams shall be concurrently supported by the Analysis_and_Classification function.

DESIGN FACTORS:

    1. Performance
        1.1 Response Time: 1 sec
        1.2 Capability: 50 * number of Class beam (KIPS)
        1.3 Relative Activity: N/A
        1.4 Speed: N/A
        1.5 Throughput:
            - Input rate:
                - # Class beams * 512 samples/1 sec * 32 bits
                - # Class operators (Class selects) * 24 bytes * 4/sec
                - (position) lat 10 bytes
                                long 10 bytes
                                heading/heading rate 10 bytes
                                roll/roll rate 10 bytes
                                pitch/pitch rate 10 bytes
                                  speed 4 bytes
                                depth rate 4 bytes
                                      * 4/sec
              - (time) 8 bytes * 4/sec
            - Output rate:

        1.6 Latency: N/A
        1.7 Efficiency: N/A
        1.8 Predictability: N/A

    2. Real-Time
        2.1 Deadlines
            2.1.1 Hard Deadlines: n/a
            2.1.2 Soft Deadline:
        2.2 Synchronization: N/A

    3. Computation/Processing Requirements
        3.1 Importance: N/A
        3.2 Usefulness: N/A
        3.3 Priority: 8
        3.4 (Computing) Portability: N/A
        3.5 Interrupt/Reset Capabilities: N/A
        3.6 Memory Space:
            Input buffer + Output buffer + Program storage

    4. Dependability
        4.1 Reliability: N/A
        4.2 Accuracy: N/A
        4.3 Fault Tolerance: N/A

4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A
        Ease of Replacement: N/A
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
4.5 Quality: N/A

5. Security
        5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    General Purpose Processor

NAME AND TITLE: **2.5.1**   **Auto_Selects_Analysis_Beam**

CHILD PROCESSES:
    None

INPUT:
Position	4/sec
Time	4/sec
Analysis_Control	

OUTPUT:
    Selected_Beams
    Auto_Class_Beams_Request

OUTPUT TRANSFORMATION:
    Auto_Class_Beams_Request <--- Position + Time.
    Selected_Beams <--- Position + Time.

PROCESS DESCRIPTION:
    - This function will automatically select tracker beams for analysis in the absence of operator selections.

DESIGN CHARACTERISTIC:
    - The tracker beam selections will be based upon the following criteria:
    (1)   Trackers with contact numbers which have not been previously been analyzed.
    (2)   Trackers with contact numbers which are classified as unknown targets.

DESIGN FACTORS:
    1. Performance
        1.1 Response Time: 1 sec
        1.2 Capability: 50 * number of Class beam (KIPS)
        1.3 Relative Activity: N/A
        1.4 Speed: N/A
        1.5 Throughput:
            - Input rate:
                - # Class beams * 512 samples/1 sec * 32 bits
                - (position) lat 10 bytes
                        long 10 bytes
                        heading/heading rate 10 bytes
                        roll/roll rate 10 bytes
                        pitch/pitch rate 10 bytes
                        speed 4 bytes

depth rate 4 bytes
* 4/sec
- (time) 8 bytes * 4/sec
- Output rate:

1.6 Latency: N/A
1.7 Efficiency: N/A
1.8 Predictability: N/A

2. Real-Time
2.1 Deadlines
2.1.1 Hard Deadlines: n/a
2.1.2 Soft Deadline:
2.2 Synchronization: N/A

3. Computation/Processing Requirements
3.1 Importance: N/A
3.2 Usefulness: N/A
3.3 Priority: 8
3.4 (Computing) Portability: N/A
3.5 Interrupt/Reset Capabilities: N/A
3.6 Memory Space:
Input buffer + Output buffer + Program storage

4. Dependability
4.1 Reliability: N/A
4.2 Accuracy: N/A
4.3 Fault Tolerance: N/A
4.4 Availability
Availability: N/A
Inherent Availability: N/A
Achieved Availability: N/A
Operational Availability: N/A
Ease of Replacement: N/A
Crash Recoverability: N/A
Computation Heavy Process Effects: N/A
4.5 Quality: N/A

5. Security
5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
General Purpose Processor

NAME AND TITLE: 2.5.2   Operator_Selects_Analysis_Beam

CHILD PROCESSES:
None

INPUT:
Class_Selects   aperiodic (2/ min)
Analysis_Control   aperiodic (2/ min)

OUTPUT:
Selected_Beams
Operator_Selected_Class_Beams_Request

OUTPUT TRANSFORMATION:
    Selected_Beams <--- Class_Selects + Analysis_Control.
    Operator_Selected_Class_Beams_Request <--- Class_Selects _Analysis_Control.

PROCESS DESCRIPTION:
    This function supports the operator in selecting tracker beams for analysis by providing a prioritized list of operative trackers and the means for assigning classification channels.

DESIGN CHARACTERISTIC:
    TBD

DESIGN FACTORS:
    1. Performance
        1.1 Response Time: 1 sec
        1.2 Capability: 50 * number of Class beam (KIPS)
        1.3 Relative Activity: N/A
        1.4 Speed: N/A
        1.5 Throughput:
            - Input rate:
                        - # Class operators (Class selects) * 24 bytes * 4/sec
                        - (position) lat 10 bytes
                                        long 10 bytes
                                        heading/heading rate 10 bytes
                                        roll/roll rate 10 bytes
                                        pitch/pitch rate 10 bytes
                                        speed 4 bytes
                                        depth rate 4 bytes
                                                * 4/sec
                    - (time) 8 bytes * 4/sec
                - Output rate:

        1.6 Latency: N/A
        1.7 Efficiency: N/A
        1.8 Predictability: N/A

    2. Real-Time
        2.1 Deadlines
                2.1.1 Hard Deadlines: n/a
                2.1.2 Soft Deadline:
        2.2 Synchronization: N/A

    3. Computation/Processing Requirements
        3.1 Importance: N/A
        3.2 Usefulness: N/A
        3.3 Priority: 8
        3.4 (Computing) Portability: N/A
        3.5 Interrupt/Reset Capabilities: N/A
        3.6 Memory Space:
                Input buffer + Output buffer + Program storage

    4. Dependability
        4.1 Reliability: N/A
        4.2 Accuracy: N/A
        4.3 Fault Tolerance: N/A
        4.4 Availability
                Availability: N/A
                Inherent Availability: N/A
                Achieved Availability: N/A
                Operational Availability: N/A
                Ease of Replacement: N/A

Crash Recoverability: N/A
Computation Heavy Process Effects: N/A
4.5 Quality: N/A

5. Security
5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
General Purpose Processor

NAME AND TITLE: **2.5.3**   **Compare_Signature**

CHILD PROCESSES:
None

INPUT:

Position	4/sec
Time	4/sec
Class_Beams	1/sec
Selected_Beams	aperiodic (2/ min)

OUTPUT:
Class_Data

OUTPUT TRANSFORMATION:
Class_Data <— Position + Time + Selected_Beams.

PROCESS DESCRIPTION:
This function compares the acoustic signature of an operator or machine selected tracker with a set of candidate target signatures stored in the system database.

DESIGN CHARACTERISTIC:
A correlation metric is calculated based upon the degree to which the stored signature matches the received signature. A list of likely candidates is developed by storing the possible candidates by the correlation metric and identifying all candidates which exceed an operator selected threshold.

DESIGN FACTORS:
1. Performance
1.1 Response Time: 1 sec
1.2 Capability: 50 * number of Class beam (KIPS)
1.3 Relative Activity: N/A
1.4 Speed: N/A
1.5 Throughput:
- Input rate:
- (position) lat 10 bytes
long 10 bytes
heading/heading rate 10 bytes
roll/roll rate 10 bytes
pitch/pitch rate 10 bytes
speed 4 bytes
depth rate 4 bytes
* 4/sec
- (time) 8 bytes * 4/sec
- Output rate:

1.6 Latency: N/A
1.7 Efficiency: N/A

1.8 Predictability: N/A

2. Real-Time
      2.1 Deadlines
            2.1.1 Hard Deadlines: n/a
            2.1.2 Soft Deadline:
      2.2 Synchronization: N/A

3. Computation/Processing Requirements
      3.1 Importance: N/A
      3.2 Usefulness: N/A
      3.3 Priority: 8
      3.4 (Computing) Portability: N/A
      3.5 Interrupt/Reset Capabilities: N/A
      3.6 Memory Space:
            Input buffer + Output buffer + Program storage

4. Dependability
      4.1 Reliability: N/A
      4.2 Accuracy: N/A
      4.3 Fault Tolerance: N/A
      4.4 Availability
            Availability: N/A
            Inherent Availability: N/A
            Achieved Availability: N/A
            Operational Availability: N/A
            Ease of Replacement: N/A
            Crash Recoverability: N/A
            Computation Heavy Process Effects: N/A
      4.5 Quality: N/A

5. Security
      5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
      General Purpose Processor

NAME AND TITLE: **2.6**   **Detection**

CHILD PROCESSES:
      None

INPUT:
      Position
      Time
      Detect_Beams
      Detect_Selects

OUTPUT:
      Detect_Data
      Gain_Control_Signal

OUTPUT TRANSFORMATION:
      Detect_Data <--- Position + Time + Detect_Beams + Detect_Selects.
      Gain_Control_Signal <--- Position + Time + Detect_Beams + Detect_Selects.

PROCESS DESCRIPTION:

- Detection Function compares the signal amplitude of each beam to a series of thresholds.
- Signal amplitude unit is converted to the corresponding threshold level (i.e. is quantized).
- The requantized data is integrated over time and formatted for display.

DESIGN CHARACTERISTIC:
- Passive detection display data shall be computed and displayed at least 4 times per second.
- Display of detection data shall be synchronized to within 100 msec of the corresponding audio data.
- Detection data will be quantized in XQ level.
- Requantized detection data is integrated over XT pre-defined time interval.
- Integrated beam data for each beam is grouped with the adjacent beam to achieve 360 degree azimuthal and 120 degree in D/E coverage in each integration period for display.

DESIGN FACTOR:
  1. Performance
        1.1 Response Time: 250 msec
        1.2 Capability: 10 * number of detection beam (KIPS)
        1.3 Relative Activity: N/A
        1.4 Speed: N/A
        1.5 Throughput:N/A
        1.6 Latency: N/A
        1.7 Efficiency: N/A
        1.8 Predictability: N/A

  2. Real-Time
        2.1 Deadlines
                2.1.1 Hard Deadlines: N/A
                2.1.2 Soft Deadline: 250 msec display update rate
        2.2 Synchronization: Display of detection data shall be synchronized to within 100 msec of the corresponding audio data.

  3. Computation/Processing Requirements
        3.1 Importance: N/A
        3.2 Usefulness: N/A
        3.3 Priority: 1
        3.4 (Computing) Portability: N/A
        3.5 Interrupt/Reset Capabilities: N/A
        3.6 Memory Space:
                Input buffer + Output buffer + Program storage

  4. Dependability
        4.1 Reliability: N/A
        4.2 Accuracy: N/A
        4.3 Fault Tolerance: N/A
        4.4 Availability
                Availability: N/A
                Inherent Availability: N/A
                Achieved Availability: N/A
                Operational Availability: N/A
                Ease of Replacement: N/A
                Crash Recoverability: N/A
                Computation Heavy Process Effects: N/A
        4.5 Quality: N/A

  5. Security
        5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
        General Purpose Processor

NAME AND TITLE: **3**   **Time_Synchronization**

CHILD PROCESSES:
>None

INPUT :
>GMT

OUTPUT:
>Time
>Sample_Synch

OUTPUT TRANSFORMATION:
>Time <— GMT
>Sample_Synch <— GMT

PROCESS DESCRIPTION:
>Time_Synchronization provides the current time through out the system and generates the synchronization pulses which coordinate the signal conditioner and the beamformer. These parameters are calculated based upon inputs from the ship's navigation system.

DESIGN CHARACTERISTIC:
>- The delivered time signal shall be accurate within 50 msec and not drift more than 10 msec over a six hour period.
>- The Sample_Synch pulse will reach the beamforming function at a 512 Hz rate +/- .005 Hz.

DESIGN FACTORS:
>1. Performance
>>1.1 Response Time: TBD
>>1.2 Capability: 10 (KIPS)
>>1.3 Relative Activity: N/A
>>1.4 Speed: N/A
>>1.5 Throughput: N/A
>>1.6 Latency: N/A
>>1.7 Efficiency: N/A
>>1.8 Predictability: N/A

>2. Real-Time
>>2.1 Deadlines
>>>2.1.1 Hard Deadlines: The delivered time signal shall be accurate within 50 msec and not drift more than 10 msec over a six hour period.

>>>The Sample_Synch pulse will reach the beamforming function at a 512 Hz rate +/- .005 Hz.

>>>2.1.2 Soft Deadline:
>>2.2 Synchronization: N/A

>3. Computation/Processing Requirements
>>3.1 Importance: N/A
>>3.2 Usefulness: N/A
>>3.3 Priority: 1
>>3.4 (Computing) Portability: N/A
>>3.5 Interrupt/Reset Capabilities: N/A
>>3.6 Memory Space:
>>>Input buffer (100 bytes + Output buffer (100 bytes+ Program storage (2 Kbytes)

>4. Dependability
>>4.1 Reliability: N/A
>>4.2 Accuracy: N/A
>>4.3 Fault Tolerance: N/A
>>4.4 Availability

Availability: N/A
Inherent Availability: N/A
Achieved Availability: N/A
Operational Availability: N/A
Ease of Replacement: N/A
Crash Recoverability: N/A
Computation Heavy Process Effects: N/A
4.5 Quality: N/A

5. Security
5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
General Purpose Processor

NAME AND TITLE: 4   **Data_Formatting_and_Option_Selection**

CHILD PROCESSES:
Detect_Data_Formatting
Detect_Decision_Making
Detect_Select_Data_Formatting
Track_Data_Formatting
Track_Decision_Making
Track_Select_Data_Formatting
Class_Data_Formatting
Class_Decision_Making
Class_Select_Data_Formatting

INPUT:
Processed_Data

OUTPUT:
Selection_Information
Analysis_Control

OUTPUT TRANSFORMATION:
Selection_Information <— Processed_Data
Analysis_Control <— Processed_Data

PROCESS DESCRIPTION:
Data formatting and option selection provides the mechanism for detection, classification, tracking, and maintenance information generated by the system to be formatted for evaluation and analyzed (either automatically or by an operator). This function also provides a mechanism whereby options selected based upon this analysis are communicated to the signal processing function.

DESIGN CHARACTERISTIC:
- Passive detection display data shall be computed and displayed at least 4 times per second.
- Display of detection data shall be synchronized to within 100 msec of the corresponding audio data.
- Detection data will be quantized in XQ level.
- Requantized detection data is integrated over XT pre-defined time interval.
- Integrated beam data for each beam is grouped with the adjacent beam to achieve 360 degree azimuthal and 120 degree in D/E coverage in each integration period for display.

DESIGN FACTOR:
1. Performance
1.1 Response Time: N/A
1.2 Capability: TBD

1.3 Relative Activity: N/A
1.4 Speed: N/A
1.5 Throughput: TBD
1.6 Latency: N/A
1.7 Efficiency: N/A
1.8 Predictability: N/A

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: TBD
        2.1.2 Soft Deadline: TBD
    2.2 Synchronization:
3. Computation/Processing Requirements
    3.1 Importance: N/A
    3.2 Usefulness: N/A
    3.3 Priority: N/a
    3.4 (Computing) Portability: N/A
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space: TBD

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A
        Ease of Replacement: N/A
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
    4.5 Quality: N/A

5. Security
    5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    TBD

NAME AND TITLE: **4.1**   **Detect_Data_Formatting**

CHILD PROCESSES:
    None

INPUT:
    Detect_Data

OUTPUT:
    Detect_Info

OUTPUT TRANSFORMATION:
    Detect_Info <--- Detect_Data

PROCESS DESCRIPTION:
    This function generates the displays for the detection of targets using the system.

DESIGN CHARACTERISTIC:

DESIGN FACTOR:
    1. Performance
        1.1 Response Time: N/A
        1.2 Capability: TBD
        1.3 Relative Activity: N/A
        1.4 Speed: N/A
        1.5 Throughput: TBD
        1.6 Latency: N/A
        1.7 Efficiency: N/A
        1.8 Predictability: N/A

    2. Real-Time
        2.1 Deadlines
            2.1.1 Hard Deadlines: TBD
            2.1.2 Soft Deadline: TBD
        2.2 Synchronization:
    3. Computation/Processing Requirements
        3.1 Importance: N/A
        3.2 Usefulness: N/A
        3.3 Priority: N/a
        3.4 (Computing) Portability: N/A
        3.5 Interrupt/Reset Capabilities: N/A
        3.6 Memory Space: TBD

    4. Dependability
        4.1 Reliability: N/A
        4.2 Accuracy: N/A
        4.3 Fault Tolerance: N/A
        4.4 Availability
            Availability: N/A
            Inherent Availability: N/A
            Achieved Availability: N/A
            Operational Availability: N/A
            Ease of Replacement: N/A
            Crash Recoverability: N/A
            Computation Heavy Process Effects: N/A
        4.5 Quality: N/A

    5. Security
        5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    TBD

NAME AND TITLE: **4.2**   **Detect_Decision_Making**

CHILD PROCESSES:
    None

INPUT:
    Detect_Info
    Sound

OUTPUT:
    Detect_Decisions

OUTPUT TRANSFORMATION:
    Detect_Decisions <--- Detect_Info + Sound.

PROCESS DESCRIPTION:
    This function represents the heuristic decision making operations typically conducted by an operator.

DESIGN CHARACTERISTIC:

DESIGN FACTOR:
    1. Performance
            1.1 Response Time: N/A
            1.2 Capability: TBD
            1.3 Relative Activity: N/A
            1.4 Speed: N/A
            1.5 Throughput: TBD
            1.6 Latency: N/A
            1.7 Efficiency: N/A
            1.8 Predictability: N/A

    2. Real-Time
            2.1 Deadlines
                    2.1.1 Hard Deadlines: TBD
                    2.1.2 Soft Deadline: TBD
            2.2 Synchronization:
    3. Computation/Processing Requirements
            3.1 Importance: N/A
            3.2 Usefulness: N/A
            3.3 Priority: N/a
            3.4 (Computing) Portability: N/A
            3.5 Interrupt/Reset Capabilities: N/A
            3.6 Memory Space: TBD

    4. Dependability
            4.1 Reliability: N/A
            4.2 Accuracy: N/A
            4.3 Fault Tolerance: N/A
            4.4 Availability
                    Availability: N/A
                    Inherent Availability: N/A
                    Achieved Availability: N/A
                    Operational Availability: N/A
                    Ease of Replacement: N/A
                    Crash Recoverability: N/A
                    Computation Heavy Process Effects: N/A
            4.5 Quality: N/A

    5. Security
            5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    TBD

NAME AND TITLE: **4.3   Detect_Select_Data_Formatting**

CHILD PROCESSES:
    None

INPUT:
>   Detect_Decision

OUTPUT:
>   Detect_Selects
>   Audio_Selects

OUTPUT TRANSFORMATION:
>   Detect_Selects <— Detect_Decision
>   Audio_Selects <— Detect_Decision

PROCESS DESCRIPTION:
>   This function generates the controls and operator entered detection data for the system.

DESIGN CHARACTERISTIC:

DESIGN FACTOR:
1. Performance
>   1.1 Response Time: N/A
>   1.2 Capability: TBD
>   1.3 Relative Activity: N/A
>   1.4 Speed: N/A
>   1.5 Throughput: TBD
>   1.6 Latency: N/A
>   1.7 Efficiency: N/A
>   1.8 Predictability: N/A

2. Real-Time
>   2.1 Deadlines
>>   2.1.1 Hard Deadlines: TBD
>>   2.1.2 Soft Deadline: TBD
>   2.2 Synchronization:
3. Computation/Processing Requirements
>   3.1 Importance: N/A
>   3.2 Usefulness: N/A
>   3.3 Priority: N/a
>   3.4 (Computing) Portability: N/A
>   3.5 Interrupt/Reset Capabilities: N/A
>   3.6 Memory Space: TBD

4. Dependability
>   4.1 Reliability: N/A
>   4.2 Accuracy: N/A
>   4.3 Fault Tolerance: N/A
>   4.4 Availability
>>   Availability: N/A
>>   Inherent Availability: N/A
>>   Achieved Availability: N/A
>>   Operational Availability: N/A
>>   Ease of Replacement: N/A
>>   Crash Recoverability: N/A
>>   Computation Heavy Process Effects: N/A
>   4.5 Quality: N/A

5. Security
>   5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:

TBD

NAME AND TITLE: **4.4**   **Track_Data_Formatting**

CHILD PROCESSES:
     None

INPUT:
     Track_Data

OUTPUT:
     Track_Info

OUTPUT TRANSFORMATION:
     Track_Info <— Track_Data

PROCESS DESCRIPTION:
     This function generates the displays for the tracking of targets using the system.

DESIGN CHARACTERISTIC:

DESIGN FACTOR:
     1. Performance
         1.1 Response Time: N/A
         1.2 Capability: TBD
         1.3 Relative Activity: N/A
         1.4 Speed: N/A
         1.5 Throughput: TBD
         1.6 Latency: N/A
         1.7 Efficiency: N/A
         1.8 Predictability: N/A

     2. Real-Time
         2.1 Deadlines
             2.1.1 Hard Deadlines: TBD
             2.1.2 Soft Deadline: TBD
         2.2 Synchronization:
     3. Computation/Processing Requirements
         3.1 Importance: N/A
         3.2 Usefulness: N/A
         3.3 Priority: N/a
         3.4 (Computing) Portability: N/A
         3.5 Interrupt/Reset Capabilities: N/A
         3.6 Memory Space: TBD

     4. Dependability
         4.1 Reliability: N/A
         4.2 Accuracy: N/A
         4.3 Fault Tolerance: N/A
         4.4 Availability
             Availability: N/A
             Inherent Availability: N/A
             Achieved Availability: N/A
             Operational Availability: N/A
             Ease of Replacement: N/A
             Crash Recoverability: N/A
             Computation Heavy Process Effects: N/A
         4.5 Quality: N/A

     5. Security

5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
        TBD

NAME AND TITLE: 4.5   **Track_Decision_Making**

CHILD PROCESSES:
        None

INPUT:
        Track_Info
        Sound

OUTPUT:
        Track_Decisions

OUTPUT TRANSFORMATION:
        Track_Decisions <— Track_Info + Sound.

PROCESS DESCRIPTION:
        This function represents the heuristic decision making operations typically conducted by an operator.

DESIGN CHARACTERISTIC:

DESIGN FACTOR:
        1. Performance
                1.1 Response Time: N/A
                1.2 Capability: TBD
                1.3 Relative Activity: N/A
                1.4 Speed: N/A
                1.5 Throughput: TBD
                1.6 Latency: N/A
                1.7 Efficiency: N/A
                1.8 Predictability: N/A

        2. Real-Time
                2.1 Deadlines
                        2.1.1 Hard Deadlines: TBD
                        2.1.2 Soft Deadline: TBD
                2.2 Synchronization:
        3. Computation/Processing Requirements
                3.1 Importance: N/A
                3.2 Usefulness: N/A
                3.3 Priority: N/a
                3.4 (Computing) Portability: N/A
                3.5 Interrupt/Reset Capabilities: N/A
                3.6 Memory Space: TBD

        4. Dependability
                4.1 Reliability: N/A
                4.2 Accuracy: N/A
                4.3 Fault Tolerance: N/A
                4.4 Availability
                        Availability: N/A
                        Inherent Availability: N/A
                        Achieved Availability: N/A

Operational Availability: N/A
Ease of Replacement: N/A
Crash Recoverability: N/A
Computation Heavy Process Effects: N/A
4.5 Quality: N/A

5. Security
5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
TBD

NAME AND TITLE: 4.6　　　Track_Select_Data_Formatting

CHILD PROCESSES:
None

INPUT:
Track_Decision

OUTPUT:
Track_Selects
Audio_Selects

OUTPUT TRANSFORMATION:
Track_Selects <— Track_Decision
Audio_Selects <— Track_Decision

PROCESS DESCRIPTION:
This function generates the controls and operator entered tracking data for the system.

DESIGN CHARACTERISTIC:

DESIGN FACTOR:
1. Performance
1.1 Response Time: N/A
1.2 Capability: TBD
1.3 Relative Activity: N/A
1.4 Speed: N/A
1.5 Throughput: TBD
1.6 Latency: N/A
1.7 Efficiency: N/A
1.8 Predictability: N/A

2. Real-Time
2.1 Deadlines
2.1.1 Hard Deadlines: TBD
2.1.2 Soft Deadline: TBD
2.2 Synchronization:
3. Computation/Processing Requirements
3.1 Importance: N/A
3.2 Usefulness: N/A
3.3 Priority: N/a
3.4 (Computing) Portability: N/A
3.5 Interrupt/Reset Capabilities: N/A
3.6 Memory Space: TBD

4. Dependability
      4.1 Reliability: N/A
      4.2 Accuracy: N/A
      4.3 Fault Tolerance: N/A
      4.4 Availability
            Availability: N/A
            Inherent Availability: N/A
            Achieved Availability: N/A
            Operational Availability: N/A
            Ease of Replacement: N/A
            Crash Recoverability: N/A
            Computation Heavy Process Effects: N/A
      4.5 Quality: N/A

5. Security
      5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
      TBD

NAME AND TITLE: 4.7   **Class_Data_Formatting**

CHILD PROCESSES:
      None

INPUT:
      Class_Data

OUTPUT:
      Class_Info

OUTPUT TRANSFORMATION:
      Class_Info <— Class_Data

PROCESS DESCRIPTION:
      This function generates the displays for the classification of targets using the system.

DESIGN CHARACTERISTIC:

DESIGN FACTOR:
      1. Performance
            1.1 Response Time: N/A
            1.2 Capability: TBD
            1.3 Relative Activity: N/A
            1.4 Speed: N/A
            1.5 Throughput: TBD
            1.6 Latency: N/A
            1.7 Efficiency: N/A
            1.8 Predictability: N/A

      2. Real-Time
            2.1 Deadlines
                  2.1.1 Hard Deadlines: TBD
                  2.1.2 Soft Deadline: TBD
            2.2 Synchronization:
      3. Computation/Processing Requirements
            3.1 Importance: N/A

3.2 Usefulness: N/A
3.3 Priority: N/a
3.4 (Computing) Portability: N/A
3.5 Interrupt/Reset Capabilities: N/A
3.6 Memory Space: TBD

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A
        Ease of Replacement: N/A
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
    4.5 Quality: N/A

5. Security
    5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    TBD

NAME AND TITLE: **4.8**   **Class_Decision_Making**

CHILD PROCESSES:
    None

INPUT:
    Class_Info
    Sound

OUTPUT:
    Class_Decisions

OUTPUT TRANSFORMATION:
    Class_Decisions <— Class_Info + Sound.

PROCESS DESCRIPTION:
    This function represents the heuristic decision making operations typically conducted by an operator.

DESIGN CHARACTERISTIC:

DESIGN FACTOR:
    1. Performance
        1.1 Response Time: N/A
        1.2 Capability: TBD
        1.3 Relative Activity: N/A
        1.4 Speed: N/A
        1.5 Throughput: TBD
        1.6 Latency: N/A
        1.7 Efficiency: N/A
        1.8 Predictability: N/A

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: TBD
        2.1.2 Soft Deadline: TBD
    2.2 Synchronization:
3. Computation/Processing Requirements
    3.1 Importance: N/A
    3.2 Usefulness: N/A
    3.3 Priority: N/a
    3.4 (Computing) Portability: N/A
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space: TBD

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A
        Ease of Replacement: N/A
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
    4.5 Quality: N/A

5. Security
    5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    TBD

NAME AND TITLE: **4.9**   **Class_Select_Data_Formatting**

CHILD PROCESSES:
    None

INPUT:
    Class_Decision

OUTPUT:
    Class_Selects
    Audio_Selects

OUTPUT TRANSFORMATION:
    Class_Selects <--- Class_Decision
    Audio_Selects <--- Class_Decision

PROCESS DESCRIPTION:
    This function generates the controls and operator entered classification data for the system.

DESIGN CHARACTERISTIC:

DESIGN FACTOR:
    1. Performance
        1.1 Response Time: N/A

1.2 Capability: TBD
1.3 Relative Activity: N/A
1.4 Speed: N/A
1.5 Throughput: TBD
1.6 Latency: N/A
1.7 Efficiency: N/A
1.8 Predictability: N/A

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: TBD
        2.1.2 Soft Deadline: TBD
    2.2 Synchronization:
3. Computation/Processing Requirements
    3.1 Importance: N/A
    3.2 Usefulness: N/A
    3.3 Priority: N/a
    3.4 (Computing) Portability: N/A
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space: TBD

4. Dependability
    4.1 Reliability: N/A
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: N/A
        Inherent Availability: N/A
        Achieved Availability: N/A
        Operational Availability: N/A
        Ease of Replacement: N/A
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
    4.5 Quality: N/A

5. Security
    5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
    TBD

NAME AND TITLE: 5   **Acoustic_Energy_Conversion**

CHILD PROCESSES:
    None

INPUT:
    Acoustic_Signal

OUTPUT:
    Electronic_Signal

OUTPUT TRANSFORMATION:
    Electronic_Signal <--- Acoustic_Signal.

PROCESS DESCRIPTION:
    Acoustic energy in the water is converted to electrical energy via a transduction process such as piezo-electric effect.

DESIGN CHARACTERISTIC:
>The conversion must be efficient in the frequency band between 10 Hz and Hz 256.

DESIGN FACTORS:
>1. Performance
>>1.1 Response Time: N/A
>>1.2 Capability: N/A
>>1.3 Relative Activity: N/A
>>1.4 Speed: N/A
>>1.5 Throughput: N/A
>>1.6 Latency: N/A
>>1.7 Efficiency: N/A
>>1.8 Predictability: N/A

>2. Real-Time
>>2.1 Deadlines
>>>2.1.1 Hard Deadline: N/A
>>>2.1.2 Soft Deadline: N/A
>>2.2 Synchronization: N/A

>3. Computation/Processing Requirements
>>3.1 Importance: N/A
>>3.2 Usefulness: N/A
>>3.3 Priority: N/A
>>3.4 (Computing) Portability: N/A
>>3.5 Interrupt/Reset Capabilities: N/A
>>3.6 Memory Space: N/A

>4. Dependability
>>4.1 Reliability: N/A
>>4.2 Accuracy: N/A
>>4.3 Fault Tolerance: N/A
>>4.4 Availability
>>>Availability: N/A
>>>Inherent Availability: N/A
>>>Achieved Availability: N/A
>>>Operational Availability: N/A
>>>Ease of Replacement: N/A
>>>Crash Recoverability: N/A
>>>Computation Heavy Process Effects: N/A
>>4.5 Quality: N/A

>5. Security
>>5.1 Level: Unclass

DESIGN RATIONALE:

COMPATIBLE RESOURCE TYPE:
>Hydrophone

Data Flow Description of the Passive Sonar System

1	Acoustic_Signal
2	Amplified_Signal
3	Analysis_Control
4	Audio_Beams
5	Audio_Beams_Request
6	Audio_Class_Selects
7	Audio_Detect_Selects
8	Audio_Selects
9	Audio_Track_Selects
10	Auto_Class_Beams_Request
11	CIC_Track
12	Class_Beams_Request
13	Class_Beams
14	Class_Data
15	Class_Info
16	Class_Selects
17	Decisions
18	Detect_Beams
19	Detect_Data
20	Detect_Info
21	Detect_Selects
22	Electronic_Signal
23	Filtered_Signal
24	FIX
25	Gain_Controlled_Signal
26	GMT
27	Operator_Selected_Class_Beams_Request
28	Position
29	Processed_Data
30	Samples
31	Sample_Synch
32	Selected_Beams
33	Selected_Information
34	Sound
35	Steering_Data
36	Time
37	Track_Beams
38	Track_Data
39	Track_Info
40	Track_Selects

NAME/TITLE:　　　**Acoustic_Signal**

STRUCTURE:

　　　　DESIGN FACTOR:
　　　　　　Type: Analog
　　　　　　Unit: N/A
　　　　　　Range: N/A
　　　　　　Increment: N/A
　　　　　　Size: N/A
　　　　　　Periodicity: N/A
　　　　　　Accuracy: N/A
　　　　　　Classification: Unclassify
　　　　　　Format/Encoding: N/A

TIMING CONSTRAINTS:

NAME/TITLE:　　　**Amplified_Signal**

STRUCTURE:

　　　　DESIGN FACTOR:
　　　　　　Type: Analog
　　　　　　Unit: N/A
　　　　　　Range: N/A
　　　　　　Increment: N/A
　　　　　　Size: N/A
　　　　　　Periodicity: N/A
　　　　　　Accuracy: N/A
　　　　　　Classification: Unclassify
　　　　　　Format/Encoding: N/A

TIMING CONSTRAINTS:

NAME/TITLE:　　　**Analysis_Control**

STRUCTURE:

　　　　DESIGN FACTOR:
　　　　　　Type: Control
　　　　　　Unit: Boolean
　　　　　　Range: 0/1
　　　　　　Increment: N/A
　　　　　　Size: 1 Byte
　　　　　　Periodicity: Aperiodic
　　　　　　Accuracy: N/A
　　　　　　Classification: Unclassify
　　　　　　Format/Encoding: C

TIMING CONSTRAINTS:

NAME/TITLE:　　　**Audio_Beams**

STRUCTURE:
　　Time + [Audio_Beam_1 | Audio_Beam_2 | Audio_Beam_3 | Audio_Beam_4]

Time
> DESIGN FACTOR:
>> Type: Data
>> Unit: hr/mim/sec
>> Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
>> Increment: (1/512) sec
>> Size: 10 Bytes
>> Periodicity: (1/512) sec
>> Accuracy: 0.0001 sec
>> Classification: Unclassify
>> Format/Encoding: HH MM SS.SSSS

Audio_Beam_1
> DESIGN FACTOR:
>> Type: Data
>> Unit: Volt
>> Range: 0 – 15 V
>> Increment: 15/(2exp32)
>> Size: 4 Bytes
>> Periodicity: (1/512) sec
>> Accuracy: 15/(2exp32)
>> Classification: Unclassify
>> Format/Encoding: VVVV

Audio_Beam_2
> DESIGN FACTOR:
>> Type: Data
>> Unit: Volt
>> Range: 0 – 15 V
>> Increment: 15/(2exp32)
>> Size: 4 Bytes
>> Periodicity: (1/512) sec
>> Accuracy: 15/(2exp32)
>> Classification: Unclassify
>> Format/Encoding: VVVV

Audio_Beam_3
> DESIGN FACTOR:
>> Type: Data
>> Unit: Volt
>> Range: 0 – 15 V
>> Increment: 15/(2exp32)
>> Size: 4 Bytes
>> Periodicity: (1/512) sec
>> Accuracy: 15/(2exp32)
>> Classification: Unclassify
>> Format/Encoding: VVVV

Audio_Beam_4
> DESIGN FACTOR:
>> Type: Data
>> Unit: Volt
>> Range: 0 – 15 V
>> Increment: 15/(2exp32)
>> Size: 4 Bytes
>> Periodicity: (1/512) sec
>> Accuracy: 15/(2exp32)
>> Classification: Unclassify
>> Format/Encoding: VVVV

TIMING CONSTRAINTS:

NAME/TITLE: **Audio_Beams_Request**

STRUCTURE:

    Time + Beam_ID_# + Beam_Type

    Time

        DESIGN FACTOR:

            Type: Data
            Unit: hr/mim/sec
            Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
            Increment: 1 sec
            Size: 8 Bytes
            Periodicity: Aperiodic
            Accuracy: 0.01 sec
            Classification: Unclassify
            Format/Encoding: HH MM SS.SS

    Beam_ID_#

        DESIGN FACTOR:

            Type: Data
            Unit: Integer
            Range: 0 – 120
            Increment: 1
            Size: 3 Bytes
            Periodicity: Aperiodic
            Accuracy: N/A
            Classification: Unclassify
            Format/Encoding: NNN

    Beam_Type

        DESIGN FACTOR:

            Type: Data
            Unit: Alphanumeric
            Range: D/T/C
            Increment: N/A
            Size: 1 Byte
            Periodicity: Aperiodic
            Accuracy: N/A
            Classification: Unclassify
            Format/Encoding: T

TIMING CONSTRAINTS:

NAME/TITLE: **Audio_Class_Selects**

STRUCTURE:

    Time + Class_Beam_ID

    Time

        DESIGN FACTOR:

            Type: Data
            Unit: hr/mim/sec
            Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
            Increment: 1 sec
            Size: 8 Bytes
            Periodicity: 250 msec
            Accuracy: 0.01 sec

              Classification: Unclassify
              Format/Encoding: HH MM SS.SS

Class_Beam_ID
        DESIGN FACTOR:
              Type: Data
              Unit: Integer
              Range: 0 – 120 (0 means not assigned)
              Increment: 1
              Size: 3 Bytes
              Periodicity: Aperiodic
              Accuracy: N/A
              Classification: Unclassify
              Format/Encoding: NNN

TIMING CONSTRAINTS:

NAME/TITLE: **Audio_Detect_Selects**

STRUCTURE:
        Time + Detect_Beam_ID

Time
        DESIGN FACTOR:
              Type: Data
              Unit: hr/mim/sec
              Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
              Increment: 1 sec
               Size: 8 Bytes
               Periodicity: 250 msec
               Accuracy: 0.01 sec
              Classification: Unclassify
              Format/Encoding: HH MM SS.SS

Detect_Beam_ID
        DESIGN FACTOR:
              Type: Data
              Unit: Integer
              Range: 0 – 120 (0 means not assigned)
              Increment: 1
              Size: 3 Bytes
              Periodicity: Aperiodic
              Accuracy: N/A
              Classification: Unclassify
              Format/Encoding: NNN

TIMING CONSTRAINTS:

NAME/TITLE: **Audio_Selects**

STRUCTURE:
        Audio_Detect_Selects | Audio_Class_Selects | Audio_Track_Selects

TIMING CONSTRAINTS:

NAME/TITLE: **Audio_Track_Selects**

STRUCTURE:
      Time + Track_Beam_ID

      Time
           DESIGN FACTOR:
                Type: Data
                Unit: hr/mim/sec
                Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
                Increment: 1 sec
                Size: 8 Bytes
                Periodicity: 250 msec
                Accuracy: 0.01 sec
                Classification: Unclassify
                Format/Encoding: HH MM SS.SS

      Track_Beam_ID
           DESIGN FACTOR:
                Type: Data
                Unit: Integer
                Range: 0 – 100 (0 means not assigned)
                Increment: 1
                Size: 3 Bytes
                Periodicity: Aperiodic
                Accuracy: N/A
                Classification: Unclassify
                Format/Encoding: NNN

TIMING CONSTRAINTS:

NAME/TITLE: **Auto_Class_Beams_Request**

STRUCTURE:
      Time + [Track_Beam_ID_Ch_1 | ..... | Track_Beam_ID_Ch_12]
      Time
           DESIGN FACTOR:
                Type: Data
                Unit: hr/mim/sec
                Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
                Increment: 1 sec
                Size: 8 Bytes
                Periodicity: 250 msec
                Accuracy: 0.01 sec
                Classification: Unclassify
                Format/Encoding: HH MM SS.SS

      Track_Beam_ID_Channel_1
           DESIGN FACTOR:
                Type: Data
                Unit: Integer
                Range: 0 – 100 (0 means not assigned)
                Increment: 1
                Size: 3 Bytes
                Periodicity: Aperiodic
                Accuracy: N/A
                Classification: Unclassify
                Format/Encoding: NNN

      •

-
-
Track_Beam_ID_Channel_12
    DESIGN FACTOR:
        Type: Data
        Unit: Integer
        Range: 0 – 100 (0 means not assigned)
        Increment: 1
        Size: 3 Bytes
        Periodicity: Aperiodic
        Accuracy: N/A
        Classification: Unclassify
        Format/Encoding: NNN

TIMING CONSTRAINTS:

NAME/TITLE: **CIC_Track**

STRUCTURE:
Time + [(CIC_Track_Beam_1_Tracker_ID + CIC_Track_Beam_1_Bearing + CIC_Track_Beam_1_Signal_to_Noise_Ratio) | ..... | (CIC_Track_Beam_100_Tracker_ID + CIC_Track_Beam_100_Bearing + CIC_Track_Beam_100_Signal_to_Noise_Ratio)]

Time
    DESIGN FACTOR:
        Type: Data
        Unit: hr/mim/sec
        Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
        Increment: 1 sec
        Size: 8 Bytes
        Periodicity: 250 msec
        Accuracy: 0.01 sec
        Classification: Unclassify
        Format/Encoding: HH MM SS.SS

CIC_Track_Beam_1_Tracker_ID
    DESIGN FACTOR:
        Type: Data
        Unit: Integer
        Range: 0 – 100 (0 means not assigned)
        Increment: 1
        Size: 3 Bytes
        Periodicity: 1 sec
        Accuracy: N/A
        Classification: Unclassify
        Format/Encoding: NNN
-
-
-
CIC_Track_Beam_100_Tracker_ID
    DESIGN FACTOR:
        Type: Data
        Unit: Integer
        Range: 0 – 100 (0 means not assigned)
        Increment: 1
        Size: 3 Bytes
        Periodicity: 1 sec
        Accuracy: N/A
        Classification: Unclassify

Format/Encoding: NNN

CIC_Track_Beam_1_Bearing
    DESIGN FACTOR:
        Type: Data
        Unit: Degree
        Range: 0 – 360
        Increment: 0.1
        Size: 4 Bytes
        Periodicity: 1 sec
        Accuracy: 0.1
        Classification: Unclassify
        Format/Encoding: DDD.D

-
-
-

CIC_Track_Beam_100_Bearing
    DESIGN FACTOR:
        Type: Data
        Unit: Degree
        Range: 0 – 360
        Increment: 0.1
        Size: 4 Bytes
        Periodicity: 1 sec
        Accuracy: 0.1
        Classification: Unclassify
        Format/Encoding: DDD.D

CIC_Track_Beam_1_Signal_to_Noise_Ratio
    DESIGN FACTOR:
        Type: Data
        Unit: dB
        Range: 0 – (+/-)35
        Increment: 0.1
        Size: 4 Bytes
        Periodicity: 1 sec
        Accuracy: 0.1
        Classification: Unclassify
        Format/Encoding: s RR.R

-
-
-

CIC_Track_Beam_100_Signal_to_Noise_Ratio
    DESIGN FACTOR:
        Type: Data
        Unit: dB
        Range: 0 – (+/-)35
        Increment: 0.1
        Size: 4 Bytes
        Periodicity: 1 sec
        Accuracy: 0.1
        Classification: Unclassify
        Format/Encoding: s RR.R

TIMING CONSTRAINTS:

NAME/TITLE: **Class_Beams_Request**

STRUCTURE:

Operator_Selected_Class_Beams_Request | Auto_Class_Beams_Request

TIMING CONSTRAINTS:

NAME/TITLE: **Class_Beams**

STRUCTURE:
         Time + [Class_Beam_1 | ..... | Class_Beam_12]
         Time
                  DESIGN FACTOR:
                           Type: Data
                           Unit: hr/mim/sec
                           Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
                           Increment: 1 sec
                           Size: 10 Bytes
                           Periodicity: (1/512) sec
                           Accuracy: 0.0001 sec
                           Classification: Unclassify
                           Format/Encoding: HH MM SS.SSSS

         Class_Beam_1
                  DESIGN FACTOR:
                           Type: Data
                           Unit: Volt
                           Range: 0 – 15
                           Increment: 15/(2exp32)
                           Size: 4 Bytes
                           Periodicity: (1/512) sec
                           Accuracy: 15/(2exp32)
                           Classification: Unclassify
                           Format/Encoding: VVVV
         •
         •
         •
         Class_Beam_12
                  DESIGN FACTOR:
                           Type: Data
                           Unit: Volt
                           Range: 0 – 15
                           Increment: 15/(2exp32)
                           Size: 4 Bytes
                           Periodicity: (1/512) sec
                           Accuracy: 15/(2exp32)
                           Classification: Unclassify
                           Format/Encoding: VVVV

TIMING CONSTRAINTS:

NAME/TITLE: **Class_Data**

STRUCTURE:
         Time + [(Ch_1_LOFAR_Gram | Ch_1_DEMON_Gram | Ch_1_Transients_Gram) | ..... | (Ch_12_LOFAR_Gram | Ch_12_DEMON_Gram | Ch_12_Transients_Gram)]

         Time
                  DESIGN FACTOR:
                           Type: Data

Unit: hr/mim/sec
Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
Increment: 1 sec
Size: 8 Bytes
Periodicity: 250 msec
Accuracy: 0.01 sec
Classification: Unclassify
Format/Encoding: HH MM SS.SS

Ch_1_LOFAR_Gram
    DESIGN FACTOR:
        Type: Data Array
        Unit: Quantized Amplitude
        Range: 0 – 7
        Increment: 1 quanta
        Size: 128 Bytes
        Periodicity: 10 sec
        Accuracy: N/A
        Classification: Unclassify
        Format/Encoding: 1x128 Bytes Array (2 bins/Byte)

•
•
•

Ch_12_LOFAR_Gram
    DESIGN FACTOR:
        Type: Data Array
        Unit: Quantized Amplitude
        Range: 0 – 7
        Increment: 1 quanta
        Size: 128 Bytes
        Periodicity: 10 sec
        Accuracy: N/A
        Classification: Unclassify
        Format/Encoding: 1x128 Bytes Array (2 bins/Byte)

Ch_1_DEMON_Gram
    DESIGN FACTOR:
        Type: Data Array
        Unit: Quantized Amplitude
        Range: 0 – 7
        Increment: 1 quanta
        Size: 128 Bytes
        Periodicity: 10 sec
        Accuracy: N/A
        Classification: Unclassify
        Format/Encoding: 1x128 Bytes Array (2 bins/Byte)

•
•
•

Ch_12_DEMON_Gram
    DESIGN FACTOR:
        Type: Data Array
        Unit: Quantized Amplitude
        Range: 0 – 7
        Increment: 1 quanta
        Size: 128 Bytes
        Periodicity: 10 sec
        Accuracy: N/A
        Classification: Unclassify
        Format/Encoding: 1x128 Bytes Array (2 bins/Byte)

Ch_1_Transient_Gram
    DESIGN FACTOR:
        Type: Data Array
        Unit: Quantized Amplitude
        Range: 0 – 7
        Increment: 1 quanta
        Size: 128 Bytes
        Periodicity: 0.1 sec
        Accuracy: N/A
        Classification: Unclassify
        Format/Encoding: 1x128 Bytes Array (2 bins/Byte)

-
-
-

Ch_12_Transient_Gram
    DESIGN FACTOR:
        Type: Data Array
        Unit: Quantized Amplitude
        Range: 0 – 7
        Increment: 1 quanta
        Size: 128 Bytes
        Periodicity: 0.1 sec
        Accuracy: N/A
        Classification: Unclassify
        Format/Encoding: 1x128 Bytes Array (2 bins/Byte)

TIMING CONSTRAINTS:

NAME/TITLE: **Class_Info**

Same as Class_Data except that the information will be displayed on screen for operators to make decision

TIMING CONSTRAINTS:

NAME/TITLE: **Class_Selects**

STRUCTURE:
    Time + Tracker_ID + Bearing + TBD

Time
    DESIGN FACTOR:
        Type: Data
        Unit: hr/mim/sec
        Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
        Increment: 1 sec
        Size: 8 Bytes
        Periodicity: Aperiodic
        Accuracy: 0.01 sec
        Classification: Unclassify
        Format/Encoding: HH MM SS.SS

Tracker_ID
    DESIGN FACTOR:
        Type: Data
        Unit: Integer
        Range: 0 – 12
        Increment: 1

Size: 3 Bytes
Periodicity: Aperiodic
Accuracy: N/A
Classification: Unclassify
Format/Encoding: TTT

Bearing
DESIGN FACTOR:
Type: Data
Unit: Degree
Range: 0 – 360
Increment: 0.1 degree
Size: 4 Bytes
Periodicity: Aperiodic
Accuracy: 0.1
Classification: Unclassify
Format/Encoding: DDD.D

TBD

TIMING CONSTRAINTS:

NAME/TITLE: **Decisions**

Operators selection of Detect_Selects, Class_Selects, Track_Selects, and Audio_Selects on screen

NAME/TITLE: **Detect_Beams**

STRUCTURE:
Time + [Detect_Beam_1 | ..... | Detect_Beam_120]

Time
DESIGN FACTOR:
Type: Data
Unit: hr/mim/sec
Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
Increment: 1 sec
Size: 10 Bytes
Periodicity: (1/512) sec
Accuracy: 0.0001 sec
Classification: Unclassify
Format/Encoding: HH MM SS.SSSS

Detect_Beam_1
DESIGN FACTOR:
Type: Data
Unit: Volt
Range: 0 – 15
Increment: 15/(2exp32)
Size: 4 Bytes
Periodicity: (1/512) sec
Accuracy: 15/(2exp32)
Classification: Unclassify
Format/Encoding: VVVV

●
●
●

Detect_Beam_120
DESIGN FACTOR:
Type: Data
Unit: Volt
Range: 0 – 15
Increment: 15/(2exp32)
Size: 4 Bytes
Periodicity: (1/512) sec
Accuracy: 15/(2exp32)
Classification: Unclassify
Format/Encoding: VVVV

TIMING CONSTRAINTS:

NAME/TITLE: **Detect_Data**

STRUCTURE:
Time + [Beam_1_amplitude | ..... | Beam_120_Amplitude]

Time
DESIGN FACTOR:
Type: Data
Unit: hr/mim/sec
Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
Increment: 1 sec
Size: 8 Bytes
Periodicity: 250 msec
Accuracy: 0.01 sec
Classification: Unclassify
Format/Encoding: HH MM SS.SS

Beam_1_Amplitude
DESIGN FACTOR:
Type: Data
Unit: Quanta
Range: 0 – 7
Increment: 1
Size: 1 Byte
Periodicity: 250 msec
Accuracy: N/A
Classification: Unclassify
Format/Encoding: A

-
-
-

Beam_120_Amplitude
DESIGN FACTOR:
Type: Data
Unit: Quanta
Range: 0 – 7
Increment: 1
Size: 1 Byte
Periodicity: 250 msec
Accuracy: N/A
Classification: Unclassify
Format/Encoding: A

TIMING CONSTRAINTS:

NAME/TITLE: **Detect_Info**

        Same as Detect_Data except that the information will be displayed on screen for operators to make decision

NAME/TITLE: **Detect_Selects**

STRUCTURE:
        Time + TBD

        Time
                DESIGN FACTOR:
                        Type: Data
                        Unit: hr/mim/sec
                        Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
                        Increment: 1 sec
                        Size: 8 Bytes
                        Periodicity: 250 msec
                        Accuracy: 0.01 sec
                        Classification: Unclassify
                        Format/Encoding: HH MM SS.SS

            TBD

TIMING CONSTRAINTS:

NAME/TITLE: **Electronic_Signal**

STRUCTURE:

                DESIGN FACTOR:
                        Type: Analog
                        Unit: N/A
                        Range: N/A
                        Increment: N/A
                        Size: N/A
                        Periodicity: N/A
                        Accuracy: N/A
                        Classification: Unclassify
                        Format/Encoding: N/A

TIMING CONSTRAINTS:

NAME/TITLE: **Filtered_Signal**

STRUCTURE:

                DESIGN FACTOR:
                        Type: Analog
                        Unit: N/A
                        Range: N/A
                        Increment: N/A
                        Size: N/A
                        Periodicity: N/A

Accuracy: N/A
Classification: Unclassify
Format/Encoding: N/A

TIMING CONSTRAINTS:

NAME/TITLE: **FIX**

STRUCTURE:
Longitude + Latitude + Heading + Roll + Pitch + Velocity + Depth

Longitude
DESIGN FACTOR:
Type: Data
Unit: deg/mim/sec
Range: 0 – 360 deg
Increment: 0.01 sec
Size: 9 Bytes
Periodicity: 62.5 msec
Accuracy: 0.01 sec
Classification: Unclassify
Format/Encoding: DDD MM SS.SS

Latitude   °
DESIGN FACTOR:
Type: Data
Unit: deg/mim/sec
Range: 0 – 90 N/S deg
Increment: 0.01 sec
Size: 9 Bytes
Periodicity: 62.5 msec
Accuracy: 0.01 sec
Classification: Unclassify
Format/Encoding: DDD MM SS.SS

Heading
DESIGN FACTOR:
Type: Data
Unit: deg
Range: 0 – 360 deg
Increment: 0.1 deg
Size: 4 Bytes
Periodicity: 62.5 msec
Accuracy: 0.1 deg
Classification: Unclassify
Format/Encoding: DDD.D

Roll
DESIGN FACTOR:
Type: Data
Unit: deg (P/S)
Range: 0 – 80 deg
Increment: 0.1 deg
Size: 4 Bytes
Periodicity: 62.5 msec
Accuracy: 0.1 deg
Classification: Unclassify
Format/Encoding: DD.D O

Pitch

DESIGN FACTOR:

Type: Data
Unit: deg (U/D)
Range: 0 – 80 deg
Increment: 0.1 deg
Size: 4 Bytes
Periodicity: 62.5 msec
Accuracy: 0.1 deg
Classification: Unclassify
Format/Encoding: DD.D O

Velocity

DESIGN FACTOR:

Type: Data
Unit: mile/hr
Range: 0 – (+/-)50
Increment: 0.1
Size: 4 Bytes
Periodicity: 62.5 msec
Accuracy: 0.1
Classification: Unclassify
Format/Encoding: s vv.v

Depth

DESIGN FACTOR:

Type: Data
Unit: feet
Range: 0 – 2000
Increment: 0.1
Size: 6 Bytes
Periodicity: 62.5 msec
Accuracy: 0.1
Classification: Unclassify
Format/Encoding: dddd.d

TIMING CONSTRAINTS:

NAME/TITLE: **Gain_Controlled_Signal**

STRUCTURE:

DESIGN FACTOR:

Type: Analog
Unit: N/A
Range: N/A
Increment: N/A
Size: N/A
Periodicity: N/A
Accuracy: N/A
Classification: Unclassify
Format/Encoding: N/A

TIMING CONSTRAINTS:

NAME/TITLE: **GMT**

STRUCTURE:

Month + Day + Hour + Minute + Second

Month

    DESIGN FACTOR:
        Type: Data
        Unit: mo
        Range: 0 – 12
        Increment: 1
        Size: 2 Bytes
        Periodicity: 1 sec
        Accuracy: 1
        Classification: Unclassify
        Format/Encoding: MO

Day

    DESIGN FACTOR:
        Type: Data
        Unit: day
        Range: 0 – 31
        Increment: 1
        Size: 2 Bytes
        Periodicity: 1 sec
        Accuracy: 1
        Classification: Unclassify
        Format/Encoding: DD

Hour

    DESIGN FACTOR:
        Type: Data
        Unit: hr
        Range: 0 – 24
        Increment: 1
        Size: 2 Bytes
        Periodicity: 1 sec
        Accuracy: 1
        Classification: Unclassify
        Format/Encoding: HH

Minute

    DESIGN FACTOR:
        Type: Data
        Unit: min
        Range: 0 – 60
        Increment: 1
        Size: 2 Bytes
        Periodicity: 1 sec
        Accuracy: 1
        Classification: Unclassify
        Format/Encoding: MM

Second

    DESIGN FACTOR:
        Type: Data
        Unit: sec
        Range: 0 – 60
        Increment: 0.0001
        Size: 6 Bytes
        Periodicity: 1 sec
        Accuracy: 0.0001
        Classification: Unclassify
        Format/Encoding: SS.SSSS

TIMING CONSTRAINTS:

NAME/TITLE: **Operator_Selected_Class_Beams_Request**

STRUCTURE:
   Time + [Track_Beam_ID_Ch_1 | ..... | Track _Beam_ID_Ch_12]

   Time
         DESIGN FACTOR:
               Type: Data
               Unit: hr/mim/sec
               Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
               Increment: 1 sec
               Size: 8 Bytes
               Periodicity: 250 msec
               Accuracy: 0.01 sec
               Classification: Unclassify
               Format/Encoding: HH MM SS.SS

   Track_Beam_ID_Ch_1
         DESIGN FACTOR:
               Type: Data
               Unit: integer
               Range: 0 – 100 (0 means not assigned)
               Increment: 1
               Size: 3 Bytes
               Periodicity: Aperiodic
               Accuracy: N/A
               Classification: Unclassify
               Format/Encoding: NNN
   •
   •
   •
   Track_Beam_ID_Ch_12
         DESIGN FACTOR:
               Type: Data
               Unit: integer
               Range: 0 – 100 (0 means not assigned)
               Increment: 1
               Size: 3 Bytes
               Periodicity: Aperiodic
               Accuracy: N/A
               Classification: Unclassify
               Format/Encoding: NNN

TIMING CONSTRAINTS:

NAME/TITLE: **Position**

STRUCTURE:
   Longitude + Latitude + Heading + Heading_Rate + Roll + Roll_Rate + Pitch + Pitch_Rate + Velocity + Acceleration + Depth + Depth_Rate

   Longitude
         DESIGN FACTOR:
               Type: Data
               Unit: deg/mim/sec

Range: 0 – 360 deg
Increment: 0.01 sec
Size: 9 Bytes
Periodicity: 62.5 msec
Accuracy: 0.01 sec
Classification: Unclassify
Format/Encoding: DDD MM SS.SS

Latitude
DESIGN FACTOR:
Type: Data
Unit: deg/mim/sec
Range: 0 – 90 N/S deg
Increment: 0.01 sec
Size: 9 Bytes
Periodicity: 62.5 msec
Accuracy: 0.01 sec
Classification: Unclassify
Format/Encoding: DDD MM SS.SS

Heading
DESIGN FACTOR:
Type: Data
Unit: deg
Range: 0 – 360 deg
Increment: 0.1 deg
Size: 4 Bytes
Periodicity: 62.5 msec
Accuracy: 0.1 deg
Classification: Unclassify
Format/Encoding: DDD.D

Heading_Rate
DESIGN FACTOR:
Type: Data
Unit: deg/sec
Range: 0 – (+/-)10
Increment: 0.1 deg/sec
Size: 4 Bytes
Periodicity: 62.5 msec
Accuracy: 0.1 deg/sec
Classification: Unclassify
Format/Encoding: s DD.D

Roll
DESIGN FACTOR:
Type: Data
Unit: deg (P/S)
Range: 0 – 80 deg
Increment: 0.1 deg
Size: 4 Bytes
Periodicity: 62.5 msec
Accuracy: 0.1 deg
Classification: Unclassify
Format/Encoding: DD.D O

Roll_Rate
DESIGN FACTOR:
Type: Data
Unit: deg/sec (P/S)
Range: 0 – 10

NSWCDD/TR-95/125

Increment: 0.1
Size: 4 Bytes
Periodicity: 62.5 msec
Accuracy: 0.1
Classification: Unclassify
Format/Encoding: DD.D O

Pitch
    DESIGN FACTOR:
        Type: Data
        Unit: deg (U/D)
        Range: 0 – 80 deg
        Increment: 0.1 deg
        Size: 4 Bytes
        Periodicity: 62.5 msec
        Accuracy: 0.1 deg
        Classification: Unclassify
        Format/Encoding: DD.D O

Pitch_Rate
    DESIGN FACTOR:
        Type: Data
        Unit: deg/sec (U/D)
        Range: 0 – 10
        Increment: 0.1
        Size: 4 Bytes
        Periodicity: 62.5 msec
        Accuracy: 0.1
        Classification: Unclassify
        Format/Encoding: DD.D O

Velocity
    DESIGN FACTOR:
        Type: Data
        Unit: mile/hr
        Range: 0 – (+/-)50
        Increment: 0.1
        Size: 4 Bytes
        Periodicity: 62.5 msec
        Accuracy: 0.1
        Classification: Unclassify
        Format/Encoding: s vv.v

Acceleration
    DESIGN FACTOR:
        Type: Data
        Unit: (mile/hr)/hr
        Range: 0 – (+/-)100
        Increment: 0.1
        Size: 5 Bytes
        Periodicity: 62.5 msec
        Accuracy: 0.1
        Classification: Unclassify
        Format/Encoding: s AAA.A

Depth
    DESIGN FACTOR:
        Type: Data
        Unit: feet
        Range: 0 – 2000
        Increment: 0.1

B-83

Size: 6 Bytes
Periodicity: 62.5 msec
Accuracy: 0.1
Classification: Unclassify
Format/Encoding: dddd.d

Depth_Rate
    DESIGN FACTOR:
        Type: Data
        Unit: feet/sec
        Range: 0 – (+/-)100
        Increment: 0.1
        Size: 5 Bytes
        Periodicity: 62.5 msec
        Accuracy: 0.1
        Classification: Unclassify
        Format/Encoding: s ddd.d

TIMING CONSTRAINTS:

NAME/TITLE: **Processed_Data**

STRUCTURE:
    Detect_Data | Class_Data | Track_Data | Sound

TIMING CONSTRAINTS:

NAME/TITLE: **Sample_Synch**

STRUCTURE:

    DESIGN FACTOR:
        Type: Data
        Unit: (1/512) sec
        Range: 0 – 2exp32
        Increment: (1/512) sec
        Size: 4 Bytes
        Periodicity: (1/512) sec
        Accuracy: (1/512) sec
        Classification: Unclassify
        Format/Encoding: SSSS

TIMING CONSTRAINTS:

NAME/TITLE: **Samples**

STRUCTURE:
    Time + [Hydro_Ch_1_Samples | ..... | Hydro_Ch_1600_Samples]

Time
    DESIGN FACTOR:
        Type: Data
        Unit: hr/mim/sec
        Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
        Increment: 0.0001 sec

Size: 10 Bytes
Periodicity: (1/512) sec
Accuracy: 0.0001 sec
Classification: Unclassify
Format/Encoding: HH MM SS.SSSS

Hydro_Ch_1_Samples
DESIGN FACTOR:
Type: Data
Unit: Volt
Range: 0 – 15
Increment: 15/(2exp32)
Size: 4 Bytes
Periodicity: Aperiodic
Accuracy: 15/(2exp32)
Classification: Unclassify
Format/Encoding: VVVV
•
•
•
Hydro_Ch_1600_Samples
DESIGN FACTOR:
Type: Data
Unit: Volt
Range: 0 – 15
Increment: 15/(2exp32)
Size: 4 Bytes
Periodicity: Aperiodic
Accuracy: 15/(2exp32)
Classification: Unclassify
Format/Encoding: VVVV

TIMING CONSTRAINTS:

NAME/TITLE: **Selected_Beams**

STRUCTURE:
Time + [Track_Beam_ID_Ch_1 | ..... | Track_Beam_ID_Ch_12]

Time
DESIGN FACTOR:
Type: Data
Unit: hr/mim/sec
Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
Increment: 1 sec
Size: 8 Bytes
Periodicity: 250 msec
Accuracy: 0.01 sec
Classification: Unclassify
Format/Encoding: HH MM SS.SS

Track_Beam_ID_Ch_1
DESIGN FACTOR:
Type: Data
Unit: integer
Range: 0 – 100 (0 means not assigned)
Increment: 1
Size: 3 Bytes

                              Periodicity: Aperiodic / 5 sec
                              Accuracy: N/A
                              Classification: Unclassify
                              Format/Encoding: NNN
    ●
    ●
    ●
Track_Beam_ID_Ch_12
                DESIGN FACTOR:
                              Type: Data
                              Unit: integer
                              Range: 0 – 100 (0 means not assigned)
                              Increment: 1
                              Size: 3 Bytes
                              Periodicity: Aperiodic / 5 sec
                              Accuracy: N/A
                              Classification: Unclassify
                              Format/Encoding: NNN

TIMING CONSTRAINTS:

NAME/TITLE: **Selected_Information**

STRUCTURE:
        Detect_Selects | Class_Selects | Track_Selects | Audio_Selects

TIMING CONSTRAINTS:

NAME/TITLE: **Sound**

STRUCTURE:

                DESIGN FACTOR:
                              Type: Analog Audio Signal
                              Unit: N/A
                              Range: 0 – 256 Hz
                              Increment: N/A
                              Size: N/A
                              Periodicity: N/A
                              Accuracy: TBD
                              Classification: Unclassify
                              Format/Encoding: N/A

TIMING CONSTRAINTS:

NAME/TITLE: **Steering_Data**

STRUCTURE:
        Time + [(Steering_Beam_1_Tracker_ID + Tracker_Bearing_Beam_1) | ..... | (Steering_Beam_100_Tracker_ID +
        Tracker_Bearing_Beam_100)]

        Time
                DESIGN FACTOR:
                              Type: Data
                              Unit: hr/mim/sec

Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
Increment: 1 sec
Size: 8 Bytes
Periodicity: 250 msec
Accuracy: 0.01 sec
Classification: Unclassify
Format/Encoding: HH MM SS.SS

Steering_Beam_1_Tracker_ID
    DESIGN FACTOR:
        Type: Data
        Unit: integer
        Range: 0 – 100 (0 means not assigned)
        Increment: 1
        Size: 3 Bytes
        Periodicity: 1 sec
        Accuracy: N/A
        Classification: Unclassify
        Format/Encoding: NNN

-
-
-

Steering_Beam_100_Tracker_ID
    DESIGN FACTOR:
        Type: Data
        Unit: integer
        Range: 0 – 100 (0 means not assigned)
        Increment: 1
        Size: 3 Bytes
        Periodicity: 1 sec
        Accuracy: N/A
        Classification: Unclassify
        Format/Encoding: NNN

Tracker_Bearing_Beam_1
    DESIGN FACTOR:
        Type: Data
        Unit: deg
        Range: 0 – 360
        Increment: 0.1
        Size: 4 Bytes
        Periodicity: 1 sec
        Accuracy: 0.1
        Classification: Unclassify
        Format/Encoding: DDD.D

-
-
-

Tracker_Bearing_Beam_100
    DESIGN FACTOR:
        Type: Data
        Unit: deg
        Range: 0 – 360
        Increment: 0.1
        Size: 4 Bytes
        Periodicity: 1 sec
        Accuracy: 0.1
        Classification: Unclassify
        Format/Encoding: DDD.D

TIMING CONSTRAINTS:

NAME/TITLE: **Time**

STRUCTURE:

Day + Hour + Minute + Second

Day

DESIGN FACTOR:

Type: Data
Unit: day
Range: 0 – 31
Increment: 1
Size: 2 Bytes
Periodicity: 1 sec
Accuracy: 1
Classification: Unclassify
Format/Encoding: DD

Hour

DESIGN FACTOR:

Type: Data
Unit: hr
Range: 0 – 24
Increment: 1
Size: 2 Bytes
Periodicity: 1 sec
Accuracy: 1
Classification: Unclassify
Format/Encoding: HH

Minute

DESIGN FACTOR:

Type: Data
Unit: min
Range: 0 – 60
Increment: 1
Size: 2 Bytes
Periodicity: 1 sec
Accuracy: 1
Classification: Unclassify
Format/Encoding: MM

Second

DESIGN FACTOR:

Type: Data
Unit: sec
Range: 0 – 60
Increment: 0.01
Size: 4 Bytes
Periodicity: 1 sec
Accuracy: 0.01
Classification: Unclassify
Format/Encoding: SS.SS

TIMING CONSTRAINTS:

NAME/TITLE: **Track_Beams**

STRUCTURE:

       Time + [Track_Beam_1 | ..... | Track_Beam_100]

    Time
           DESIGN FACTOR:
                Type: Data
                Unit: hr/mim/sec
                Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
                Increment: 1 sec
                Size: 10 Bytes
                Periodicity: (1/512) sec
                Accuracy: 0.0001 sec
                Classification: Unclassify
                Format/Encoding: HH MM SS.SSSS

    Track_Beam_1
           DESIGN FACTOR:
                Type: Data
                Unit: Volt
                Range: 0 – 15
                Increment: 15/(2exp32)
                Size: 4 Bytes
                Periodicity: (1/512) sec
                Accuracy: 15/(2exp32)
                Classification: Unclassify
                Format/Encoding: VVVV

    ●
    ●
    ●

    Track_Beam_100
           DESIGN FACTOR:
                Type: Data
                Unit: Volt
                Range: 0 – 15
                Increment: 15/(2exp32)
                Size: 4 Bytes
                Periodicity: (1/512) sec
                Accuracy: 15/(2exp32)
                Classification: Unclassify
                Format/Encoding: VVVV

TIMING CONSTRAINTS:

NAME/TITLE: **Track_Data**

STRUCTURE:

    Time + [(Track_Beam_1_Tracker_ID + Track_Beam_1_Bearing + Track_Beam_1_Signal_to_Noise_Ratio) | ..... | (Track_Beam_100_Tracker_ID + Track_Beam_100_Bearing + Track_Beam_100_Signal_to_Noise_Ratio)]

    Time
           DESIGN FACTOR:
                Type: Data
                Unit: hr/mim/sec
                Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
                Increment: 1 sec
                Size: 8 Bytes
                Periodicity: 250 msec
                Accuracy: 0.01 sec
                Classification: Unclassify

Format/Encoding: HH MM SS.SS

Track_Beam_1_Tracker_ID
      DESIGN FACTOR:
          Type: Data
          Unit: integer
          Range: 0 – 100 (0 means not assigned)
          Increment: 1
          Size: 3 Bytes
          Periodicity: 1 sec
          Accuracy: N/A
          Classification: Unclassify
          Format/Encoding: NNN

-
-
-

Track_Beam_100_Tracker_ID
      DESIGN FACTOR:
          Type: Data
          Unit: integer
          Range: 0 – 100 (0 means not assigned)
          Increment: 1
          Size: 3 Bytes
          Periodicity: 1 sec
          Accuracy: N/A
          Classification: Unclassify
          Format/Encoding: NNN

Track_Beam_1_Bearing
      DESIGN FACTOR:
          Type: Data
          Unit: deg
          Range: 0 – 360
          Increment: 0.1
          Size: 4 Bytes
          Periodicity: 1 sec
          Accuracy: 0.1
          Classification: Unclassify
          Format/Encoding: DDD.D

-
-
-

Track_Beam_100_Bearing
      DESIGN FACTOR:
          Type: Data
          Unit: deg
          Range: 0 – 360
          Increment: 0.1
          Size: 4 Bytes
          Periodicity: 1 sec
          Accuracy: 0.1
          Classification: Unclassify
          Format/Encoding: DDD.D

Track_Beam_1_Signal_to_Noise_Ratio
      DESIGN FACTOR:
          Type: Data
          Unit: dB
          Range: 0 – (+/-)35
          Increment: 0.1
          Size: 4 Bytes

Periodicity: 1 sec
Accuracy: 0.1
Classification: Unclassify
Format/Encoding: s RR.R

-
-
-

Track_Beam_100_Signal_to_Noise_Ratio
DESIGN FACTOR:
Type: Data
Unit: dB
Range: 0 – (+/-)35
Increment: 0.1
Size: 4 Bytes
Periodicity: 1 sec
Accuracy: 0.1
Classification: Unclassify
Format/Encoding: s RR.R

TIMING CONSTRAINTS:

NAME/TITLE: **Track_Info**

Same as Track_Data except that the information will be displayed on screen for operators to make decision

NAME/TITLE: **Track_Select**

STRUCTURE:
Time + Tracker_ID + Bearing + TBD

Time
DESIGN FACTOR:
Type: Data
Unit: hr/mim/sec
Range: 0 – 24 hr/0 – 60 mim/0 – 60 sec
Increment: 1 sec
Size: 8 Bytes
Periodicity: Aperiodic
Accuracy: 0.01 sec
Classification: Unclassify
Format/Encoding: HH MM SS.SS

Tracker_ID
DESIGN FACTOR:
Type: Data
Unit: integer
Range: 0 – 100 (0 means not assigned)
Increment: 1
Size: 3 Bytes
Periodicity: Aperiodic
Accuracy: N/A
Classification: Unclassify
Format/Encoding: TTT

Bearing
DESIGN FACTOR:
Type: Data

Unit: deg
Range: 0 – 360
Increment: 0.1
Size: 4 Bytes
Periodicity: Aperiodic
Accuracy: 0.1
Classification: Unclassify
Format/Encoding: DDD.D

TIMING CONSTRAINTS:

SECTION 4

IMPLEMENTATION CAPTURE VIEW

Summary of the Implementation Capture View Method

The Implementation Capture View documents the hardware, software and human resources which represent a particular embodiment of the system under design. A multi-level mapping is also established which relates the system functions (with associated data flow and behavior contained in the Functional and Behavioral Capture Views) to the resources in the Implementation Capture View. The key results of generating the Implementation Capture View is the hardware, software and humanware architecture descriptions for the system under design. These descriptions represent the principal products of the systems engineering effort and establish a baseline for the detailed design of the system hardware, software and operating concept. Information that was generated in the development of the Implementation Capture View is summarized in the following paragraphs.

Resource Library. A resource library is a repository for all candidate resources that can be used or built throughout the design process. The resource library exists in the form of a database where all candidate resources are categorized from a very general class to a specific type. While the descriptions and performance capabilities of the off-the-shelf products are documented accordingly to their manufacture specifications, the "to-be-designed" or modified resources are listed by their expected values including their design constraints.

Multi-Level Function-Resource Mapping. The multi-level function-resource mapping provides a mechanism for allocating functions to resource types and ultimately for mapping those functions indirectly to specific resources. The mapping establishes a strong link between the logical architecture and the resource architecture and requires that (1) every function be fully implemented in resources, and (2) every resource be traceable back to a required function (or a derived system service task). For the passive sonar system three levels of mapping are established.

The first layer of the function-resource mapping is an assignment of the system functions to generalized resource classes. Each function is mapped to one of these generalized resource classes which is then further refined by specifying a certain resource type from that class. For example, the beamforming function is mapped to the general class of Programmable Hardware & Software, which is further specified as special purpose custom beamformer hardware with beamformer microcode software.

The second layer of the function-resource mapping establishes a set of implementation tasks which will be performed by the specific resource classes or types. Many of these implementation tasks are directly related to the functions and may represent an implementation specific functional partitioning, grouping or some combination of the two which reflects the intended implementation. Other implementation tasks are created to provide required implementation specific system services such as CPU operating systems, database managers and network executives. This layer of the mapping provides the systems engineer with a mechanism for repartitioning the functional decomposition for implementation without directly modifying the Functional Capture View.

The third layer of function-resource mapping is the allocation of the implementation tasks to specific resources. At this level the description of the candidate resources are detailed enough such that a framework for the system design can be constructed. In this layer of the mapping individual resources are identified by hardware unit number, software/humanware task and human operator for each implementation task. Many implementation tasks require both

hardware and software resources. The static allocation of implementation tasks to specific combinations of hardware and software represents an over simplification of the system's hardware-software mapping. In a system with alternate program load options or in systems where software tasks are dynamically allocated to hardware this level of the mapping simply represents a particular instance of the possible software hardware mapping.

Hardware Architecture. The hardware architecture is captured in the form of a data base with an entry for each hardware system resource, using a format similar to the resource library format illustrated in Figure 2-1. In addition to the fields provided in the resource library, the hardware architecture data base includes information on the selection rational and requirements traceability for each hardware resource in the system, the installed location (both in physical terms and in terms of the interconnection topology with other resources), and a description of any messages sent and received by the hardware which are specific to that hardware and not due to the software running on that hardware. (Messages sent or received by the software which run on the hardware are described in the software architecture.) Additional fields are also provided for each design factor to allow required, budgeted and other categories of interest to be captured.

Software Architecture. The software architecture also exists in the form of a database which includes a description of the various software modules of the system. Each module is described in terms of the processing and algorithms implemented, selected design factors addressing throughput requirements, memory requirements, etc., and a description of the messages sent and received by the module. The software architecture description is also intended to contain other information typically included in a SRS as defined by the DOD-STD-2167A DID #DIMCCR 80025A. The software architecture can be represented using various graphical forms including a listing of source code modules with calling relationships, message flow between modules, etc. The software architecture database contains the necessary information to construct these system software representations and to support modeling of system performance.

Humanware Architecture. The humanware architecture describes the number and type of personnel (i.e., training and experience levels) required to man the system under various operating conditions. When human function is a large part of the system under design, the organizational chart is considered part of this architecture. Organizational breakdown and informational interchanges between different departments are captured in a format similar to one described in the capture of hardware or software. Figure 2-5. illustrates a human resource architecture of the passive sonar system and an example description of one particular operator (i.e., detection operator).

Passive Sonar System Resource Library

      1. Sparc 1E CPU Card
      2. Detection Software Package
      3. Track Software Package

NAME/TITLE
MOTOROLA SPARC 1E CPU CARD

MANUFACTURE
Motorola

TYPE
General Purpose Computer Hardware

DESCRIPTION
Provides general purpose computing capability including CISC processor with floating point coprocessor.

SELECTION RATIONAL
TBS

DESIGN FACTORS
1. Performance:
    1.1 Response Time: f(software processing load)
    1.2 Capability:
        1.2.1 CPU Processor Type: SPARC
        1.2.2 Bus Architecture: VME
        1.2.3 CPU Word Size: 32 bit
        1.2.4 Address Bus Size: 32 bit
        1.2.5 Data Bus Size: 32 bit
        1.2.6 Interrupt/Reset Capabilities: TBD
        1.2.7 Memory Space: 16 Megabyte
    1.3 Relative Activity:
        1.3.1 CPU Utilization: f(software processing load)
        1.3.2 Interface Utilization: f(software processing load)
            Serial Port 1: f(software processing load)
            Serial Port 2: f(software processing load)
            Parallel Port 1: f(software processing load)
            Parallel Port 2: f(software processing load)
            Network Port 1: f(software processing load)
            Network Port 2: f(software processing load)
    1.4 Speed:
        1.4.1 CPU Clock Speed: 25 Mhz
        1.4.2 MIPS: TBD
        1.4.3 FLOPS: TBD
    1.5 Throughput: f(software processing load)
    1.6 Latency: f(software processing load)
    1.7 Efficiency: f(software processing load)
    1.8 Predictability: TBD

2. Real-Time:
    2.1 Deadlines:
        2.1.1 Hard Deadlines: N/A
        2.1.2 Soft Deadline: N/A
    2.2 Synchronization: TBD

3. Computation/Processing Requirements:
    3.1 Importance: TBD (e.g., single point of failure, multiple redundant)
    3.2 Usefulness: TBD (e.g., general purpose or specific use)
    3.3 Priority: N/A (this design factor applies to functions and software processes)
    3.4 (Computing) Portability: N/A (this design factor applies to software processes)

4. Dependability:
    4.1 Reliability:
        4.1.1 MTBF = 16,000 Hrs
        4.1.2 MTTF = TBD

4.2 Accuracy: TBD
4.3 Fault Tolerance:
      4.3.1 CPU: Single point of failure
      4.3.2 FPU: Failure result in reduction of FLOPS rating to 5% of specification
      4.3.3 Communication ports: each port can fail independently.
                              4.4 Availability:
                4.4.1 Availability: TBD
                4.4.2 Inherent Availability: TBD
                4.4.3 Achieved Availability: TBD
                4.4.4 Operational Availability: TBD
                4.4.5 Maintainability: MTTR = 30 min.
                4.4.6 Crash Recoverability:
                        Warm start: TBD
                        Cold start: TBD
      4.5 Quality: 2/1000 defects

5. Security:
      5.1 Level: Unclass

6. Physical requirements:
      6.1 Size: full height 6U card
      6.2 Weight: 2.4 lbs
      6.3 Ruggedness: best commercial practice
      6.4 Energy:
            6.4.1 Energy consumption: 12 Watts
            6.4.2 Energy dissipation: TBD
      6.5 Operating environment:
            6.5.1 Temperature: -10 to 70 0C
            6.5.2 Humidity: 0 to 85% (non condensing)
            6.5.3 Vibration: TBD
            6.5.4 Electromagnetic radiation: TBD

7. Financial Requirements:
      7.1 Cost to Develop: N/A
      7.2 Cost to Prototype: N/A
      7.3 Cost to Produce: N/A
      7.4 Cost to Test: N/A
      7.5 Cost to Purchase: $ 2,400.00
      7.6 Cost to Operate:
      7.7 Cost to Maintain:
      7.8 Cost to Repair: TBD
      7.9 Cost to Include Security Capability:

8. Time Projected:
      8.1 Estimated Time to Develop: N/A
      8.2 Estimated Time to Prototype: N/A
      8.3 Estimated Time to Produce: N/A
      8.4 Estimated Time to Test: N/A
      8.5 Estimated Time to Purchase:
      8.6 Estimated Time to Operate:
      8.7 Estimated Time to Maintain:
      8.8 Estimated Time to Repair:
      8.9 Estimated Time to Include Security Capability:

9. Life Cycle

COMPONENTS
      TBS

INTERFACES
      Two high speed digital interfaces (Ethernet)

INSTALLED LOCATION
TBD

NAME/TITLE: **Detection Package**

TYPE:
    Software

DESCRIPTION:
- This SW compares the signal amplitude of each beam (n to n+19) to a series of thresholds
- Signal amplitude is converted to the corresponding threshold level (i.e. is quantized)
- The requantized data is integrated over time and formatted for display

DESIGN FACTORS:
    Performance
        Processing Required: 0.22 MIPS
        Memory Required: 60 KBytes
        Priority: 3
        Response Time: 250 msec

SUBTASK:
    None

INPUT MESSAGES:
    Detect_Selects_Msg
    Time_Msg
    Position_Msg
    Detect_Beams_Msg (Beam n to n+19)

OUTPUT MESSAGES
    Detect_Data_Msg (Beam n to n+19)

NAME/TITLE: **Track Package**

TYPE:
    Software

TASK DESCRIPTION:
    This task will process Track_Beams_Msg (Beam n to n+20) to:
- Determine the estimate bearing for each assigner tracker
- Calculate beam steering coefficients to recenter each Track Beam on the estimated target position
- Compute the bearing rate for each tracker by a least squares fit of the bearing data over the most recent 30 sec interval

    Trackers are initiated on bearings identified by the Track_Selects_Msg parameters

DESIGN FACTORS:
    Performance
        Processing Required: 0.22 MIPS
        Memory Required: 60 KBytes
        Priority: 4
        Response Time: 1000 msec

SUBTASK:
    None

INPUT MESSAGES:
  Track_Selects_Msg
  Time_Msg
  Position_Msg
  Track_Beams_Msg (Beam n to n+19)

OUTPUT MESSAGES
  Track_Data_Msg (Beam n to n+19)
  CIC_Track_Msg (Beam n to n+19)
  Steering_Data_Msg (Beam n to n+19)

The Allocation of Functions to Resources

Function	General Resource Class	Specific Resource Type	Task ID	Task Description	Task Allocation HW Unit or Operator	Task Allocation SW Task or Process	SW Task Relocatablity Compatible Hardware Resource Types	Processing Required (MIPS)	Memory Required (KBytes)
Acou Energy Conv	Non Prog HW	Hyd array	HW Task 101	Acou Energy Conv	Hyd Array	N/A	N/A	N/A	N/A
Signal Conditioning	Prog HW & SW	Custom Signal Cond SC Microcode/CSCI 1	HW/SW Task 301	Sig Cond Hyd 1-400	SC #1	SW Task 301	Not relocateable	Special	Special
			HW/SW Task 302	Sig Cond Hyd 401-800	SC #2	SW Task 302	Not relocateable	Special	Special
			HW/SW Task 303	Sig Cond Hyd 801-1200	SC #3	SW Task 303	Not relocateable	Special	Special
			HW/SW Task 304	Sig Cond Hyd 1201-1600	SC #4	SW Task 304	Not relocateable	Special	Special
Beamforming	Prog HW & SW	Custom Beamformer BF Microcode/CSCI 2	HW/SW Task 401	Form Fixed Det Beams	BF #1	SW Task 305	BF#1 or BF#2	Special	Special
			HW/SW Task 402	Form Steerable Beams	BF #2	SW Task 306	BF#1 or BF#2	Special	Special
Detection	Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 3	HW/SW Task 101	Det Beams 1-20	CPU #1	SW Task 101	CPU #1 thru CPU #8	0.18	60
			HW/SW Task 102	Det Beams 21-40	CPU #1	SW Task 102	CPU #1 thru CPU #8	0.18	60
			HW/SW Task 103	Det Beams 42-60	CPU #1	SW Task 103	CPU #1 thru CPU #8	0.18	60
			HW/SW Task 104	Det Beams 61-80	CPU #2	SW Task 104	CPU #1 thru CPU #8	0.18	60
			HW/SW Task 105	Det Beams 80-100	CPU #2	SW Task 105	CPU #1 thru CPU #8	0.18	60
			HW/SW Task 106	Det Beams 101-120	CPU #2	SW Task 106	CPU #1 thru CPU #8	0.18	60
			HW/SW Task 107	Det Management	CPU #2	SW Task 106	CPU #1 thru CPU #8	0.18	60
Tracking	Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 4	HW/SW Task 111	Trk Beams 1-20	CPU #3	SW Task 111	CPU #1 thru CPU #8	0.22	60
			HW/SW Task 112	Trk Beams 21-40	CPU #3	SW Task 112	CPU #1 thru CPU #8	0.22	60
			HW/SW Task 113	Trk Beams 41-60	CPU #3	SW Task 113	CPU #1 thru CPU #8	0.22	60
			HW/SW Task 114	Trk Beams 61-80	CPU #4	SW Task 114	CPU #1 thru CPU #8	0.22	60
			HW/SW Task 115	Trk Beams 81-100	CPU #4	SW Task 115	CPU #1 thru CPU #8	0.22	60
Analysis & Classification	Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 5	HW/SW Task 121	Analysis Ch 1-4	CPU #5	SW Task 121	CPU #1 thru CPU #8	0.44	85
			HW/SW Task 122	Analysis Ch 5-8	CPU #6	SW Task 122	CPU #1 thru CPU #8	0.44	
			HW/SW Task 123	Analysis Ch 9-12	CPU #7	SW Task 123	CPU #1 thru CPU #8	0.24	
			HW/SW Task 124	Analysis Mgmt	CPU #8	SW Task 124	CPU #1 thru CPU #8	0.44	85 53
Audio	Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 6	HW/SW Task 131	Operator #1 Audio	DC #1	SW Task 131	DC #1 thru DC #4	0.1	35
			HW/SW Task 132	Operator #2 Audio	DC #2	SW Task 132	DC #1 thru DC #4	0.1	35
			HW/SW Task 133	Operator #3 Audio	DC #3	SW Task 133	DC #1 thru DC #4	0.1	35
			HW/SW Task 134	Operator #4 Audio	DC #4	SW Task 134	DC #1 thru DC #4	0.1	35
Stabilization	Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 7	HW/SW Task 141	Stabilization	CPU #5	SW Task 141	CPU #1 thru CPU #8	0.12	22
Time Sync	Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 8	HW/SW Task 151	Time Sync	CPU #6	SW Task 151	CPU #1 thru CPU #8	0.09	22
Data Formatting & Option Selection	Prog HW & SW	Display Console Ada Code/CSCI 9	HW/SW Task 161	Detection C&D	DC #1	SW Task 161	DC #1 thru DC #4	0.48	150
			HW/SW Task 162	Track C&D	DC #2	SW Task 162	DC #1 thru DC #4	0.48	150
			HW/SW Task 163	Analysis C&D	DC #3	SW Task 163	DC #1 thru DC #4	0.48	150
			HW/SW Task 164	Back up C&D	DC #4	SW Task 164	DC #1 thru DC #4	0.41	150
	Human	ST 3	Hum Task 001	Detection Operator	OP #1	Hum Task 01	OP #1 thru OP #4		
			Hum Task 002	Track Operator	OP #2	Hum Task 02	OP #1 thru OP #4		
			Hum Task 003	Analysis Operator	OP #3	Hum Task 03	OP #1 thru OP #4		
			Hum Task 004	Maintenence Operator	OP #4	Hum Task 04	OP #1 thru OP #4		
	Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 10	HW/SW Task 001	CPU #1 Op System	CPU #1	SW Task 001	Not relocateable	0.05	35
			HW/SW Task 002	CPU #2 Op System	CPU #2	SW Task 002	Not relocateable	0.05	35
			HW/SW Task 003	CPU #3 Op System	CPU #3	SW Task 003	Not relocateable	0.05	35
			HW/SW Task 004	CPU #4 Op System	CPU #4	SW Task 004	Not relocateable	0.05	35
			HW/SW Task 005	CPU #5 Op System	CPU #5	SW Task 005	Not relocateable	0.05	35
			HW/SW Task 006	CPU #6 Op System	CPU #6	SW Task 006	Not relocateable	0.05	35
			HW/SW Task 007	CPU #7 Op System	CPU #7	SW Task 007	Not relocateable	0.05	35
			HW/SW Task 008	CPU #8 Op System	CPU #8	SW Task 008	Not relocateable	0.05	35
			HW/SW Task 009	DC #1 Op System	DC #1	SW Task 009	Not relocateable	0.05	35
			HW/SW Task 00A	DC #2 Op System	DC #2	SW Task 00A	Not relocateable	0.05	35
			HW/SW Task 00B	DC #3 Op System	DC #3	SW Task 00B	Not relocateable	0.05	35
			HW/SW Task 00C	DC #4 Op System	DC #4	SW Task 00C	Not relocateable	0.05	35
			HW/SW Task 00D	System Exec	CPU #1	SW Task 00D	CPU #1 thru CPU #8	0.11	48
	Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 11	HW/SW Task 171	CPU #1 Net Node	CPU #1	SW Task 171	Not relocateable	0.03	18
			HW/SW Task 172	CPU #2 Net Node	CPU #2	SW Task 172	Not relocateable	0.03	18
			HW/SW Task 173	CPU #3 Net Node	CPU #3	SW Task 173	Not relocateable	0.03	18
			HW/SW Task 174	CPU #4 Net Node	CPU #4	SW Task 174	Not relocateable	0.03	18
			HW/SW Task 175	CPU #5 Net Node	CPU #5	SW Task 175	Not relocateable	0.03	18
			HW/SW Task 176	CPU #6 Net Node	CPU #6	SW Task 176	Not relocateable	0.03	18
			HW/SW Task 177	CPU #7 Net Node	CPU #7	SW Task 177	Not relocateable	0.03	18
			HW/SW Task 178	CPU #8 Net Node	CPU #8	SW Task 178	Not relocateable	0.03	18
			HW/SW Task 179	DC #1 Net Node	DC #1	SW Task 179	Not relocateable	0.03	18
			HW/SW Task 17A	DC #2 Net Node	DC #2	SW Task 17A	Not relocateable	0.03	18
			HW/SW Task 17B	DC #3 Net Node	DC #3	SW Task 17B	Not relocateable	0.03	18
			HW/SW Task 17C	DC #4 Net Node	DC #4	SW Task 17C	Not relocateable	0.03	18
			HW/SW Task 17D	Network Exec	CPU #2	SW Task 17D	CPU #1 thru CPU #8	0.14	48
	Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 12	HW/SW Task 181	CPU #1 DB User	CPU #1	SW Task 181	Not relocateable	0.06	34
			HW/SW Task 182	CPU #2 DB User	CPU #2	SW Task 182	Not relocateable	0.06	34
			HW/SW Task 183	CPU #3 DB User	CPU #3	SW Task 183	Not relocateable	0.06	34

		HW/SW Task 184	CPU #4	DB User	CPU #4	SW Task 184	Not relocateable	0.06	34
		HW/SW Task 185	CPU #5	DB User	CPU #5	SW Task 185	Not relocateable	0.06	34
		HW/SW Task 186	CPU #6	DB User	CPU #6	SW Task 186	Not relocateable	0.06	34
		HW/SW Task 187	CPU #7	DB User	CPU #7	SW Task 187	Not relocateable	0.06	34
		HW/SW Task 188	CPU #8	DB User	CPU #8	SW Task 188	Not relocateable	0.06	34
		HW/SW Task 189	DC#1	DB User	DC#1	SW Task 189	Not relocateable	0.06	34
		HW/SW Task 18A	DC#2	DB User	DC#2	SW Task 18A	Not relocateable	0.06	34
		HW/SW Task 18B	DC#3	DB User	DC#3	SW Task 18B	Not relocateable	0.06	34
		HW/SW Task 18C	DC#4	DB User	DC#4	SW Task 18C	Not relocateable	0.06	34
		HW/SW Task 18D	Data Base Mgr #1		CPU #4	SW Task 18D	Not relocateable	0.34	124
		HW/SW Task 18E	Data Base Mgr #2		CPU #8	SW Task 18E	Not relocateable	0.34	124
Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 13	HW/SW Task 191	CPU #1	Local PM	CPU #1	SW Task 191	Not relocateable	0.03	26
		HW/SW Task 192	CPU #2	Local PM	CPU #2	SW Task 192	Not relocateable	0.03	26
		HW/SW Task 193	CPU #3	Local PM	CPU #3	SW Task 193	Not relocateable	0.03	26
		HW/SW Task 194	CPU #4	Local PM	CPU #4	SW Task 194	Not relocateable	0.03	26
		HW/SW Task 195	CPU #5	Local PM	CPU #5	SW Task 195	Not relocateable	0.03	26
		HW/SW Task 196	CPU #6	Local PM	CPU #6	SW Task 196	Not relocateable	0.03	26
		HW/SW Task 197	CPU #7	Local PM	CPU #7	SW Task 197	Not relocateable	0.03	26
		HW/SW Task 198	CPU #8	Local PM	CPU #8	SW Task 198	Not relocateable	0.03	26
		HW/SW Task 199	DC#1	Local PM	DC#1	SW Task 199	Not relocateable	0.03	26
		HW/SW Task 19A	DC#2	Local PM	DC#2	SW Task 19A	Not relocateable	0.03	26
		HW/SW Task 19B	DC#3	Local PM	DC#3	SW Task 19B	Not relocateable	0.03	26
		HW/SW Task 19C	DC#4	Local PM	DC#4	SW Task 19C	Not relocateable	0.03	26
		HW/SW Task 19D	PM Manager		CPU #3	SW Task 19D	CPU #1 thru CPU #8	0.13	46
Prog HW & SW	SPARC 1E Processor Ada Code/CSCI 14	HW/SW Task 201	CPU #1	Local FL	CPU #1	SW Task 201	Not relocateable	0.01	22
		HW/SW Task 202	CPU #2	Local FL	CPU #2	SW Task 202	Not relocateable	0.01	22
		HW/SW Task 203	CPU #3	Local FL	CPU #3	SW Task 203	Not relocateable	0.01	22
		HW/SW Task 204	CPU #4	Local FL	CPU #4	SW Task 204	Not relocateable	0.01	22
		HW/SW Task 205	CPU #5	Local FL	CPU #5	SW Task 205	Not relocateable	0.01	22
		HW/SW Task 206	CPU #6	Local FL	CPU #6	SW Task 206	Not relocateable	0.01	22
		HW/SW Task 207	CPU #7	Local FL	CPU #7	SW Task 207	Not relocateable	0.01	22
		HW/SW Task 208	CPU #8	Local FL	CPU #8	SW Task 208	Not relocateable	0.01	22
		HW/SW Task 209	DC#1	Local FL	DC#1	SW Task 209	Not relocateable	0.01	22
		HW/SW Task 20A	DC#2	Local FL	DC#2	SW Task 20A	Not relocateable	0.01	22
		HW/SW Task 20B	DC#3	Local FL	DC#3	SW Task 20B	Not relocateable	0.01	22
		HW/SW Task 20C	DC#4	Local FL	DC#4	SW Task 20C	Not relocateable	0.01	22
		HW/SW Task 20D	FL Manager		CPU #4	SW Task 20D	CPU #1 thru CPU #8	0.08	46

Candidate Hardware Resource Architecture

PASSIVE SONAR SYSTEM

SIGNAL CONDITIONER UNIT 1001

DRAWER 1001-1

SPARC 1E
MULITPLEXER
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to

DRAWER 1001-2

SPARC 1E
MULITPLEXER
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to

DRAWER 1001-3

SPARC 1E
MULITPLEXER
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to
5 CH. AMP/FILTER/A to

DRAWER 1001-4

COOLING FANS
- 12 VOLT POWER SUPPLY
5 VOLT POWER SUPPLY
12 VOLT POWER SUPPLY

SIGNAL CONDITIONER UNIT 1002

DRAWER 1002-1

SPARC 1E
MULTIPLEXER
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D

DRAWER 1002-2

SPARC 1E
MULTIPLEXER
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D

DRAWER 1002-3

SPARC 1E
MULTIPLEXER
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D

DRAWER 1002-4

COOLING FANS
-12 VOLT POWER SUPPLY
5 VOLT POWER SUPPLY
12 VOLT POWER SUPPLY

SIGNAL CONDITIONER UNIT 1004

DRAWER 1004-1

SPARC 1E
MULTIPLEXER
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D

DRAWER 1004-2

SPARC 1E
MULTIPLEXER
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D

DRAWER 1004-3

SPARC 1E
MULTIPLEXER
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D
5 CH. AMP/FILTER/A to D

DRAWER 1004-4

COOLING FANS
- 12 VOLT POWER SUPPLY
5 VOLT POWER SUPPLY
12 VOLT POWER SUPPLY

BEAMFORMER UNIT 1101

DRAWER 1101-1

BACKPLANE 1101-1-2
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR

BACKPLANE 1101-1-1
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR

DRAWER 1101-2

BACKPLANE 1101-2-1
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR

DRAWER 1101-3

BACKPLANE 1101-3-1
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR

DRAWER 1101-4
- COOLING FANS
- -12 VOLT POWER SUPPLY
- 5 VOLT POWER SUPPLY
- 12 VOLT POWER SUPPLY

BEAMFORMER UNIT 1102

DRAWER 1102-1

BACKPLANE 1102-1-2
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR

BACKPLANE 1102-1-1
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR

DRAWER 1102-2

BACKPLANE 1102-2-1
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR

DRAWER 1102-3

BACKPLANE 1102-3-1
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR
- ARRAY PROCESSOR

DRAWER 1102-4
- 12 VOLT POWER SUPPLY
- 5 VOLT POWER SUPPLY
- - 12 VOLT POWER SUPPLY
- COOLING FANS

SIGNAL PROCESSOR / APPLICATION PROCESSOR UNIT 1201

DRAWER 1201-1

BACKPLANE 1201-1-3
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET

BACKPLANE 1201-1-2
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET

BACKPLANE 1201-1-1

DRAWER 1201-2

BACKPLANE 1201-2-3
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET

BACKPLANE 1201-2-2
- SPARC 1E
- 16 MB MEMORY
- 16 MB MEMORY
- DUAL CH. ETHERNET

BACKPLANE 1201-2-1

DRAWER 1201-3
- 1/4" TAPE DISK DRIVE
- 1.2 GB DISK DRIVE
- 1.2 GB DISK DRIVE
- 1.2 GB DISK DRIVE

DRAWER 1201-4
- COOLING FANS
- -12 VOLT POWER SUPPLY
- 5 VOLT POWER SUPPLY
- 12 VOLT POWER SUPPLY

SIGNAL PROCESSOR / APPLICATION PROCESSOR UNIT 1202

DISPLAY CONSOLE UNIT 1301

DRAWER 1301-1

19" COLOR CRT WITH TOUCH SCREEN

DRAWER 1301-2

19" COLOR CRT WITH TOUCH SCREEN

DRAWER 1301-3

BACKPLANE 1301-3-3

| SPARC 1E |
| 16 MB MEMORY |
| 16 MB MEMORY |
| ETHERNET |
| DISPLAY PROCESSOR |

BACKPLANE 1301-3-2

| SPARC 1E |
| 16 MB MEMORY |
| 16 MB MEMORY |
| ETHERNET |
| DISPLAY PROCESSOR |

BACKPLANE 1301-3-1

| SPARC 1E |
| 16 MB MEMORY |
| 16 MB MEMORY |
| DUAL CH. ETHERNET |
| SERIAL I/O |

DRAWER 1301-4

| 12 VOLT POWER SUPPLY |
| 5 VOLT POWER SUPPLY |
| - 12 VOLT POWER SUPPLY |
| COOLING FANS |

**DISPLAY CONSOLE
UNIT 1302**

DRAWER 1302-1

19" COLOR CRT
WITH TOUCH SCREEN

DRAWER 1302-2

19" COLOR CRT
WITH TOUCH SCREEN

DRAWER 1302-3

BACKPLANE 1302-3-3

SPARC 1E
16 MB MEMORY
16 MB MEMORY
ETHERNET
DISPLAY PROCESSOR

BACKPLANE 1302-3-2

SPARC 1E
16 MB MEMORY
16 MB MEMORY
ETHERNET
DISPLAY PROCESSOR

BACKPLANE 1302-3-1

SPARC 1E
16 MB MEMORY
16 MB MEMORY
DUAL CH. ETHERNET
SERIAL I/O

DRAWER 1302-4

12 VOLT
POWER SUPPLY

5 VOLT
POWER SUPPLY

- 12 VOLT
POWER SUPPLY

COOLING
FANS

DISPLAY CONSOLE
UNIT 1303

DRAWER 1303-1

19" COLOR CRT
WITH TOUCH SCREEN

DRAWER 1303-2

19" COLOR CRT
WITH TOUCH SCREEN

DRAWER 1303-3

BACKPLANE 1303-3-3

DISPLAY PROCESSOR
ETHERNET
16 MB MEMORY
16 MB MEMORY
SPARC 1E

BACKPLANE 1303-3-2

DISPLAY PROCESSOR
ETHERNET
16 MB MEMORY
16 MB MEMORY
SPARC 1E

BACKPLANE 1303-3-1

SERIAL I/O
DUAL CH. ETHERNET
16 MB MEMORY
16 MB MEMORY
SPARC 1E

DRAWER 1303-4

COOLING
FANS

- 12 VOLT
POWER SUPPLY

5 VOLT
POWER SUPPLY

12 VOLT
POWER SUPPLY

DISPLAY CONSOLE
UNIT 1304

DRAWER 1304-1

19" COLOR CRT
WITH TOUCH SCREEN

DRAWER 1304-2

19" COLOR CRT
WITH TOUCH SCREEN

DRAWER 1304-3

BACKPLANE 1304-3-3
SPARC 1E
16 MB MEMORY
16 MB MEMORY
ETHERNET
DISPLAY PROCESSOR

BACKPLANE 1304-3-2
SPARC 1E
16 MB MEMORY
16 MB MEMORY
ETHERNET
DISPLAY PROCESSOR

BACKPLANE 1304-3-1
SPARC 1E
16 MB MEMORY
16 MB MEMORY
DUAL CH. ETHERNET
SERIAL I/O

DRAWER 1304-4

COOLING FANS

- 12 VOLT POWER SUPPLY

5 VOLT POWER SUPPLY

12 VOLT POWER SUPPLY

NAME/TITLE
Beamformer

TYPE
Special Purpose Hardware

DESCRIPTION
Samples from the signal conditioners are delayed, scaled, and summed to form 120 fixed spatial beams for detection beam processing. Data from hydrophones are used to compute each beam.

SELECTION RATIONAL
TBS

DESIGN FACTORS
TBS

COMPONENTS
TBS

INTERFACES
- 6 High speed digital ports (ethernet)
Input Messages
Position_Msg
Time_Msg
Samples_Msg_(Hyd_n_to_n+399)
Samples_Msg_(Hyd_n+400_to_n+799)
Samples_Msg_(Hyd_n+800_to_n+1199)
Samples_Msg_(Hyd_n+1200_to_n+1599)
Output Messages
Sample_Sync_Msg
Detect_Beams_Msg

INSTALLED LOCATION

Beamformer #1 is connected to each of the four signal conditioners via point-to-point ethernet interfaces and to Ethernet Network #1.

NAME/TITLE
Display Console Resource

TYPE
General Purpose Computer Display and Control Hardware

DESCRIPTION
Provides general purpose computer display and control capability including two high performance display processors with 19" color CRT's, keyboard, trackball, and touch screen interfaces. The display console also has an imbedded One MIP computer which communicates with the display processors via direct memory access (DMA) channel.

SELECTION RATIONAL
TBS

DESIGN FACTORS
1. Performance
1.1 Response Time: TBD
1.2 Capability: one million 32 bit instructions per second CPU plus two display processors with 250,000 vector per second drawing speed.
1.3 Relative Activity: TBD
1.4 Speed: 25 Mhz processor clock speed
1.5 Throughput: TBD
1.6 Latency: TBD

1.7 Efficiency: N/A
1.8 Predictability: TBD

2. Real-Time
2.1 Deadlines
2.1.1 Hard Deadlines: N/A
2.1.2 Soft Deadline: N/A
2.2 Synchronization: TBD

3. Computation/Processing Requirements
3.1 Importance: TBD
3.2 Usefulness: TBD
3.3 Priority: TBD
3.4 (Computing) Portability: TBD
3.5 Interrupt/Reset Capabilities: TBD
3.6 Memory Space: one Megabyte

4. Dependability
4.1 Reliability: TBD
4.2 Accuracy: TBD
4.3 Fault Tolerance: TBD
4.4 Availability
Availability: TBD
Inherent Availability: TBD
Achieved Availability: TBD
Operational Availability: TBD
Ease of Replacement: TBD
Crash Recoverability: TBD
Computation Heavy Process Effects: TBD
4.5 Quality: TBD

5. Security
5.1 Level: Unclass

COMPONENTS
TBS
INTERFACES

- Two high speed digital interfaces (Ethernet)

INSTALLED LOCATION
Connected to Safenet Network

NAME/TITLE
Ethernet Network

TYPE
General Purpose Computer Networking Hardware

DESCRIPTION
Provides general purpose computer networking capability across fiberoptic links. Effective network bandwidth for a multiple connection network is 4 megabits per second.

SELECTION RATIONAL
TBS

DESIGN FACTORS
1. Performance
1.1 Response Time: TBD
1.2 Capability: 10 megabits per second raw transfer rate with an effective capability of 4 megabits per second.

1.3 Relative Activity: N/A
1.4 Speed: TBD
1.5 Throughput: TBD
1.6 Latency: TBD
1.7 Efficiency: TBD
1.8 Predictability: TBD

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: N/A
        2.1.2 Soft Deadline: N/A
    2.2 Synchronization: N/A

3. Computation/Processing Requirements
    3.1 Importance: TBD
    3.2 Usefulness: TBD
    3.3 Priority:N/A
    3.4 (Computing) Portability: TBD
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space: N/A

4. Dependability
    4.1 Reliability: TBD
    4.2 Accuracy: TBD
    4.3 Fault Tolerance: TBD
    4.4 Availability
        Availability: TBD
        Inherent Availability: TBD
        Achieved Availability: TBD
        Operational Availability: TBD
        Ease of Replacement: TBD
        Crash Recoverability: TBD
        Computation Heavy Process Effects: TBD
    4.5 Quality: TBD

5. Security
    5.1 Level: Unclass

COMPONENTS
TBS

INTERFACES
Fiber optic cable with terminators and "T" connectors

INSTALLED LOCATION

NAME/TITLE
Ethernet Point-to-Point High Speed Serial Interface Resource

TYPE
General Purpose Computer Interface Hardware

DESCRIPTION
Provides general purpose point-to-point high speed computer interface capability across fiberoptic links. Effective interface bandwidth for a point-to-point connection is 7 megabits per second.

SELECTION RATIONAL
TBS

DESIGN FACTORS

1. Performance
    1.1 Response Time: TBD
    1.2 Capability: 10 megabits per second raw transfer rate with an effective capability of 7 megabits per second.
    1.3 Relative Activity: N/A
    1.4 Speed: TBD
    1.5 Throughput: TBD
    1.6 Latency: TBD
    1.7 Efficiency: TBD
    1.8 Predictability: TBD

2. Real-Time
    2.1 Deadlines
        2.1.1 Hard Deadlines: N/A
        2.1.2 Soft Deadline: N/A
    2.2 Synchronization: N/A

3. Computation/Processing Requirements
    3.1 Importance: TBD
    3.2 Usefulness: TBD
    3.3 Priority:N/A
    3.4 (Computing) Portability: TBD
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space: N/A

4. Dependability
    4.1 Reliability: TBD
    4.2 Accuracy: TBD
    4.3 Fault Tolerance: TBD
    4.4 Availability
        Availability: TBD
        Inherent Availability: TBD
        Achieved Availability: TBD
        Operational Availability: TBD
        Ease of Replacement: TBD
        Crash Recoverability: TBD
        Computation Heavy Process Effects: TBD
    4.5 Quality: TBD

5. Security
    5.1 Level: Unclass

COMPONENTS
    TBS
INTERFACES

    - fiber optic cable with in-line connectors

INSTALLED LOCATION
    TBD

NAME/TITLE
    MOTOROLA SPARC 1E CPU CARD

MANUFACTURE
    Motorola

TYPE
    General Purpose Computer Hardware

DESCRIPTION
> Provides general purpose computing capability including CISC processor with floating point coprocessor.

SELECTION RATIONAL
> TBS

DESIGN FACTORS
1. Performance:
   1.1 Response Time: f(software processing load)
   1.2 Capability:
      1.2.1 CPU Processor Type: SPARC
      1.2.2 Bus Architecture: VME
      1.2.3 CPU Word Size: 32 bit
      1.2.4 Address Bus Size: 32 bit
      1.2.5 Data Bus Size: 32 bit
      1.2.6 Interrupt/Reset Capabilities: TBD
      1.2.7 Memory Space: 16 Megabyte
   1.3 Relative Activity:
      1.3.1 CPU Utilization: f(software processing load)
      1.3.2 Interface Utilization: f(software processing load)
         Serial Port 1: f(software processing load)
         Serial Port 2: f(software processing load)
         Parallel Port 1: f(software processing load)
         Parallel Port 2: f(software processing load)
         Network Port 1: f(software processing load)
         Network Port 2: f(software processing load)
   1.4 Speed:
      1.4.1 CPU Clock Speed: 25 Mhz
      1.4.2 MIPS: TBD
      1.4.3 FLOPS: TBD
   1.5 Throughput: f(software processing load)
   1.6 Latency: f(software processing load)
   1.7 Efficiency: f(software processing load)
   1.8 Predictability: TBD

2. Real-Time:
   2.1 Deadlines:
      2.1.1 Hard Deadlines: N/A
      2.1.2 Soft Deadline: N/A
   2.2 Synchronization: TBD

3. Computation/Processing Requirements:
   3.1 Importance: TBD (e.g., single point of failure, multiple redundant)
   3.2 Usefulness: TBD (e.g., general purpose or specific use)
   3.3 Priority: N/A (this design factor applies to functions and software processes)
   3.4 (Computing) Portability: N/A (this design factor applies to software processes)

4. Dependability:
   4.1 Reliability:
      4.1.1 MTBF = 16,000 Hrs
      4.1.2 MTTF = TBD
   4.2 Accuracy: TBD
   4.3 Fault Tolerance:
      4.3.1 CPU: Single point of failure
      4.3.2 FPU: Failure result in reduction of FLOPS rating to 5% of specification
      4.3.3 Communication ports: each port can fail independently.
         4.4 Availability:
         4.4.1 Availability: TBD
         4.4.2 Inherent Availability: TBD
         4.4.3 Achieved Availability: TBD
         4.4.4 Operational Availability: TBD

4.4.5 Maintainability: MTTR = 30 min.
4.4.6 Crash Recoverability:
Warm start: TBD
Cold start: TBD
4.5 Quality: 2/1000 defects

5. Security:
5.1 Level: Unclass

6. Physical requirements:
6.1 Size: full height 6U card
6.2 Weight: 2.4 lbs
6.3 Ruggedness: best commercial practice
6.4 Energy:
6.4.1 Energy consumption: 12 Watts
6.4.2 Energy dissipation: TBD
6.5 Operating environment:
6.5.1 Temperature: -10 to 70 0C
6.5.2 Humidity: 0 to 85% (non condensing)
6.5.3 Vibration: TBD
6.5.4 Electromagnetic radiation: TBD

7. Financial Requirements:
7.1 Cost to Develop: N/A
7.2 Cost to Prototype: N/A
7.3 Cost to Produce: N/A
7.4 Cost to Test: N/A
7.5 Cost to Purchase: $ 2,400.00
7.6 Cost to Operate:
7.7 Cost to Maintain:
7.8 Cost to Repair: TBD
7.9 Cost to Include Security Capability:

8. Time Projected:
8.1 Estimated Time to Develop: N/A
8.2 Estimated Time to Prototype: N/A
8.3 Estimated Time to Produce: N/A
8.4 Estimated Time to Test: N/A
8.5 Estimated Time to Purchase:
8.6 Estimated Time to Operate:
8.7 Estimated Time to Maintain:
8.8 Estimated Time to Repair:
8.9 Estimated Time to Include Security Capability:

9. Life Cycle

COMPONENTS
TBS

INTERFACES
Two high speed digital interfaces (Ethernet)

INSTALLED LOCATION
TBD

NAME/TITLE
TR-155 Hydrophone

TYPE

Special Purpose Hardware

DESCRIPTION
Piezo-electric device for converting acoustic energy in water into electrical signals.

SELECTION RATIONAL

DESIGN FACTORS

1. Performance
    1.1 Response Time: N/A
    1.2 Capability:
        1.2.1 Resonate Frequency: TBD
        1.2.2 Usable Frequency Range: TBD
        1.2.3 Beam Pattern: TBD
        1.2.4 Acoustic Efficiency: TBD
        1.2.5 Input Power: TBD
        1.2.6 Operating Depth: TBD
    1.3 Relative Activity: N/A
    1.4 Speed: N/A
    1.5 Throughput: N/A
    1.6 Latency: N/A
    1.7 Efficiency: TBD
    1.8 Predictability: N/A

2. Real-Time
    2.1 Deadlines: N/A
    2.2 Synchronization: N/A

3. Computation/Processing Requirements
    3.1 Importance: N/A
    3.2 Usefulness: N/A
    3.3 Priority: N/A
    3.4 (Computing) Portability: N/A
    3.5 Interrupt/Reset Capabilities: N/A
    3.6 Memory Space: N/A

4. Dependability
    4.1 Reliability: TBD
    4.2 Accuracy: N/A
    4.3 Fault Tolerance: N/A
    4.4 Availability
        Availability: TBD
        Inherent Availability: TBD
        Achieved Availability: TBD
        Operational Availability: TBD
        Ease of Replacement: TBD
        Crash Recoverability: N/A
        Computation Heavy Process Effects: N/A
    4.5 Quality: TBD

5. Security
    5.1 Level: Unclass

COMPONENTS
None

INTERFACES
Analog signal cable shielded with two conductors and molded connector.

INSTALLED LOCATION
    Part of Hydrophone Array

NAME/TITLE
    Signal Conditioner

TYPE
    Special Purpose Hardware

DESCRIPTION
    Provides amplification, filtering and A/D conversion and interface connections for 400 hydrophones.

SELECTION RATIONAL
    TBS

DESIGN FACTORS
    1. Performance
        1.1 Response Time: 512, 32 bit samples per sec per hydrophone
        1.2 Capability: TBD
        1.3 Relative Activity: N/A
        1.4 Speed: TBD
        1.5 Throughput: TBD
        1.6 Latency: TBD
        1.7 Efficiency: TBD
        1.8 Predictability: TBD

    2. Real-Time
        2.1 Deadlines
            2.1.1 Hard Deadlines: TBD
            2.1.2 Soft Deadline: TBD
        2.2 Synchronization: The signal conditioning and beamforming funcitons must be synchronized within .0005 sec to support a 512 sample per sec rate.

    3. Computation/Processing Requirements
        3.1 Importance: TBD
        3.2 Usefulness: TBD
        3.3 Priority:TBD
        3.4 (Computing) Portability: TBD
        3.5 Interrupt/Reset Capabilities: TBD
        3.6 Memory Space: TBD

    4. Dependability
        4.1 Reliability: TBD
        4.2 Accuracy: TBD
        4.3 Fault Tolerance: TBD
        4.4 Availability
            Availability: TBD
            Inherent Availability: TBD
            Achieved Availability: TBD
            Operational Availability: TBD
            Ease of Replacement: TBD
            Crash Recoverability: TBD
            Computation Heavy Process Effects: TBD
        4.5 Quality: TBD

    5. Security
        5.1 Level: Unclass

COMPONENTS
    - preamplifier

- filter
- amplifier/gain control
- 32 bit D/A converter

INTERFACES
- 400 analog hydrophone signal inputs

- Two high speed digital interfaces (Ethernet)
    Input Messages
        Sample_Sync_Msg
        Gain_Signal_Msg
    Output Messages
        Samples_Msg_(Hyd_n_to_n+399)

INSTALLED LOCATION

Candidate Software Architecture

The software architecture of this example was captured using Teamwork's. It is important to note that in this section the bubbles represent software tasks and the flows represent messages (or software function calls). This should not be confused with the functions and data captured using Teamwork's in the Functional/Behavioral Capture View.

Data Flow Diagram of the Passive Sonar System Software

DFD	0	PASSIVE_SONAR_SYSTEM_SW
DFD	1	Application Function Software
DFD	1.2	CSCI 1 through 6 Signal Processing
DFD	1.2.1	CSCI 2 Beamforming
DFD	1.2.2	CSCI 4 Tracking
DFD	1.2.3	CSCI 6 Audio
DFD	1.2.4	CSCI 1 Signal Conditioning
DFD	1.2.5	CSCI 5 Analysis and Classification
DFD	1.2.6	CSCI 3 Detection
DFD	1.3	CSCI 9 Data Formatting and Option Selection
DFD	2	Support Function Software
DFD	2.1	CSCI 10 Operating System
DFD	2.2	CSCI 11 Network
DFD	2.2.2	Network Nodes
DFD	2.3	CSCI 12 Data Base
DFD	2.3.1	Data Base User
DFD	2.3.2	Data Base Manager
DFD	2.4	CSCI 13 Local PM
DFD	2.5	CSCI 14 Local FL

0;2
Passive_Sonar_System_SW

Electronic_Signal_Msg

FIX_Msg

GMT_Msg

CIC_Track_Msg

Application
Function
Software

1

Support_Application_Msg

Support
Function
Software

2

1;2
Application Function Software

CSCI 7
Stabilization
SW Task 141

.1

Support_Application_Msg

FIX_Msg

Position_Msg

CSCI 9
Data Formatting
and Option
Selection

.3

Support_Application_Msg

Selected_Information_Msg

CSCI
1 through 6
Signal
Processing

.2

CIC_Track_Msg

Processed_Data_Msg

Analysis_Control_Msg

Support_Application_Msg

Electronic_Signal_Msg

CSCI 8
Time Sync
SW Task 151

.4

Support_Application_Msg

GMT_Msg

Time_Msg

Sample_Synch_Msg

1.2;3
CSCI 1 through 6 Signal Processing

Gain_Control_Signal_Msg

Detect_Data_Msg

Support_Application_Msg

Position_Msg

Time_Msg

Support_Application_Msg

Detect_Selects_Msg

Time_Msg

Position_Msg

Electronic_Signal_Msg

Sample_Synch_Msg

Support_Application_Msg

CSCI 1
Signal
Conditioning
.4

Samples_Msg

CSCI 2
Beamforming
.1

Detect_Beams_Msg

CSCI 3
Detection
.6

Class_Beams_Request_Msg

Audio_Beams_Request_Msg

Steering_Data_Msg

Track_Beams_Msg

Class_Beams_Msg

Audio_Beams_Msg

Time_Msg

Position_Msg

CSCI 6
Audio
.3

Sound_Msg

Audio_Selects_Msg

Support_Application_Msg

Time_Msg

Position_Msg

CSCI 4
Tracking
.2

Track_Selects_Msg

Track_Data_Msg

CIC_Track_Msg

Support_Application_Msg

Time_Msg

Position_Msg

Analysis_Control_Msg

Class_Selects_Msg

Class_Data_Msg

Support_Application_Msg

CSCI 5
Analysis
and
Classification
.5

1.2.1;1
CSCI 2 Beamforming

Audio_Beams_Request_Msg_Op_1

Audio_Beams_Request_Msg_Op_4

Position_Msg

Samples_Msg

Time_Msg

Support_Application_Msg

SW Task 401
Form Fixed
Det. Beams

.1

Audio_Beams_Msg_Op_1

Audio_Beams_Msg_Op_4

Detect_Beams_Msg

Audio_Beams_Request_Msg_Op_2

Audio_Beams_Request_Msg_Op_3

Audio_Beams_Request_Msg_Op_4

Class_Beams_Request_Msg

Steering_Data_Msg

Position_Msg

Samples_Msg

Time_Msg

Support_Application_Msg

SW Task 402
Form
Steerable Beams

.2

Audio_Beams_Msg_Op_2

Audio_Beams_Msg_Op_3

Audio_Beams_Msg_Op_4

Class_Beams_Msg

Track_Beams_Msg

1.2.2;1
CSCI 4 Tracking

Track_Beams_Msg_Beam_1_to_20

Time_Msg

Position_Msg

Track_Selects_Msg

Support_Application_Msg

Track_Data_Msg_Beam_1_to_20

CIC_Track_Msg_Beam_1_to_20

Steering_Data_Msg_Beam_1_to_20

SW TASK 111 TRK. BEAM 1-20 .1

Track_Beams_Msg_Beam_21_to_40

Time_Msg

Position_Msg

Track_Selects_Msg

Support_Application_Msg

Track_Data_Msg_Beam_21_to_40

CIC_Track_Msg_Beam_21_to_40

Steering_Data_Msg_Beam_21_to_40

SW TASK 112 TRK. BEAM 21-40 .2

Track_Beams_Msg_Beam_41_to_60

Time_Msg

Position_Msg

Track_Selects_Msg

Support_Application_Msg

Track_Data_Msg_Beam_41_to_60

CIC_Track_Msg_Beam_41_to_60

Steering_Data_Msg_Beam_41_to_60

SW TASK 113 TRK. BEAM 41-60 .3

Track_Beams_Msg_Beam_61_to_80

Time_Msg

Position_Msg

Track_Selects_Msg

Support_Application_Msg

Track_Data_Msg_Beam_61_to_80

CIC_Track_Msg_Beam_61_to_80

Steering_Data_Msg_Beam_61_to_80

SW TASK 114 TRK. BEAM 61-80 .4

Track_Beams_Msg_Beam_81_to_100

Time_Msg

Position_Msg

Track_Selects_Msg

Support_Application_Msg

Track_Data_Msg_Beam_81_to_100

CIC_Track_Msg_Beam_81_to_100

Steering_Data_Msg_Beam_81_to_100

SW TASK 115 TRK. BEAM 81-100 .5

1.2.3;1
CSCI 6 Audio

Support_Application_Msg

Audio_Beams_Msg_Op_1

Audio_Selects_Msg_Op_1

Position_Msg

Time_Msg

Audio_Beams_Request_Msg_Op_1

Sound_Msg_Op_1

SW Task 131
Operator 1
Audio

.1

Support_Application_Msg

Audio_Beams_Msg_Op_2

Audio_Selects_Msg_Op_2

Position_Msg

Time_Msg

Audio_Beams_Request_Msg_Op_2

Sound_Msg_Op_2

SW Task 132
Operator 2
Audio

.2

Support_Application_Msg

Audio_Beams_Msg_Op_3

Audio_Selects_Msg_Op_3

Position_Msg

Time_Msg

Audio_Beams_Request_Msg_Op_3

Sound_Msg_Op_3

SW Task 133
Operator 3
Audio

.3

Support_Application_Msg

Audio_Beams_Msg_Op_4

Audio_Selects_Msg_Op_4

Position_Msg

Time_Msg

Audio_Beams_Request_Msg_Op_4

Sound_Msg_Op_4

SW Task 134
Operator 4
Audio

.4

NOTE:
Operator 4 is the backup position. It can be assigned
to perform detection or track or class

1.2.4;1
CSCI 1 Signal Conditioning

Support_Application_Msg

Gain_Control_Signal_Msg

Sample_Synch_Msg

Electronic_Signal_Msg_
Hyd_Ch_1_to_400

Samples_Msg_
Hyd_Ch_1_to_400

SW Task 301
Sig. Cond.
Hyd. 1-400

.1

Support_Application_Msg

Gain_Control_Signal_Msg

Sample_Synch_Msg

Electronic_Signal_Msg_
Hyd_Ch_401_to_800

Samples_Msg_
Hyd_Ch_401_to_800

SW Task 302
Sig. Cond.
Hyd. 401-800

.2

Support_Application_Msg

Gain_Control_Signal_Msg

Sample_Synch_Msg

Electronic_Signal_Msg_
Hyd_Ch_801_to_1200

Samples_Msg_
Hyd_Ch_801_to_1200

SW Task 303
Sig. Cond.
Hyd. 801-1200

.3

Support_Application_Msg

Gain_Control_Signal_Msg

Sample_Synch_Msg

Electronic_Signal_Msg_
Hyd_Ch_1201_to_1600

Samples_Msg_
Hyd_Ch_1201_to_1600

SW Task 304
Sig. Cond.
Hyd. 1201-1600

.4

1.2.5;3
CSCI 5 Analysis and Classification

Class_Beams_Request_Msg

Class_Data_Msg

Analysis_Control_Msg

Position_Msg

Time_Msg

Class_Beams_Msg_Ch_1_to_4

Class_Selects_Msg_Ch_1_to_4

Support_Application_Msg

SW Task 121
Analysis
Ch. 1-4

.1

Class_Data_Msg_Ch_1_to_4

**Class_Beams_Request_
Msg_Ch_1_to_4**

Analysis_Control_Msg

Position_Msg

Time_Msg

Class_Beams_Msg_Ch_5_to_8

Class_Selects_Msg_Ch_5_to_8

Support_Application_Msg

SW Task 122
Analysis
Ch. 5-8

.2

Class_Data_Msg_Ch_5_to_8

**Class_Beams_Request_
Msg_Ch_5_to_8**

SW Task 124
Analysis Mgmt

.4

Analysis_Control_Msg

Position_Msg

Time_Msg

Class_Beams_Msg_Ch_9_to_12

Class_Selects_Msg_Ch_9_to_12

Support_Application_Msg

SW Task 123
Analysis
Ch. 9-12

.3

Class_Data_Msg_Ch_9_to_12

**Class_Beams_Request_
Msg_Ch_9_to_12**

1.2.6;4
CSCI 3 Detection

Detect_Beams_Msg_Beam_1_to_20

Detect_Selects_Msg

Time_Msg

Position_Msg

Support_Application_Msg

Detect_Data_Msg_Beam_1_to_20

**SW TASK 101
DET. BEAM
1-20

.1**

Gain_Control_Signal_Msg
Beam_1_to_20

Detect_Beams_Msg_Beam_21_to_40

Detect_Selects_Msg

Time_Msg

Position_Msg

Support_Application_Msg

Detect_Data_Msg_Beam_21_to_40

**SW TASK 102
DET. BEAM
21-40

.2**

Gain_Control_Signal_Msg
Beam_21_to_40

Detect_Beams_Msg_Beam_41_to_60

Detect_Selects_Msg

Time_Msg

Position_Msg

Support_Application_Msg

Detect_Data_Msg_Beam_41_to_60

**SW TASK 103
DET. BEAM
41-60

.3**

Gain_Control_Signal_Msg
Beam_41_to_60

Detect_Beams_Msg_Beam_61_to_80

Detect_Selects_Msg

Time_Msg

Position_Msg

Support_Application_Msg

Detect_Data_Msg_Beam_61_to_80

**SW TASK 104
DET. BEAM
61-80

.4**

Gain_Control_Signal_Msg
Beam_61_to_80

Detect_Beams_Msg_Beam_81_to_100

Detect_Selects_Msg

Time_Msg

Position_Msg

Support_Application_Msg

Detect_Data_Msg_Beam_81_to_100

**SW TASK 105
DET. BEAM
81-100

.5**

Gain_Control_Signal_Msg
Beam_81_to_100

Detect_Beams_Msg_Beam_101_to_120

Detect_Selects_Msg

Time_Msg

Position_Msg

Support_Application_Msg

Detect_Data_Msg_Beam_101_to_120

**SW TASK 106
DET. BEAM
101-120

.6**

Gain_Control_Signal_Msg
Beam_101_to_120

**SW TASK 107
DET.
MANAGEMENT

.7**

Gain_Control_
Signal_Msg

1.3;1
CSCI 9 Data Formatting and Option Selection

Support_Application_Msg

Detect_Data_Msg

Sound_Msg_Op_1

Sound_Msg_Op_4

SW Task 161
Detection
C&D

.1

Analysis_Control_Msg

Detect_Selects_Msg

Audio_Selects_Msg_Op_1

Audio_Selects_Msg_Op_4

Support_Application_Msg

Track_Data_Msg

Sound_Msg_Op_2

Sound_Msg_Op_4

SW Task 162
Track
C&D

.2

Analysis_Control_Msg

Track_Selects_Msg

Audio_Selects_Msg_Op_2

Audio_Selects_Msg_Op_4

Support_Application_Msg

Class_Data_Msg

Sound_Msg_Op_3

Sound_Msg_Op_4

SW Task 163
Class
C&D

.3

Analysis_Control_Msg

Class_Selects_Msg

Audio_Selects_Msg_Op_3

Audio_Selects_Msg_Op_4

Support_Application_Msg

Processed_Data_Msg

SW Task 164
Backup
C&D

.4

Selected_Information_Msg

Analysis_Control_Msg

2;1
Support Function Software

2.1;1
CSCI 10 Operating System

2.2;1
CSCI 11 Network

OP_Net_Msg

DB_Net_Msg

PM_Net_Msg

FL_Net_Msg

Net_App_Msg

SW Task 17D
Network Exec.

.1

Netexec_Comd_Msg

Netnodes_Status_Msg

FL_Mgr_Net_Supp_Msg

PM_Mgr_Net_Supp_Msg

Sysexe_Net_Supp_Msg

Network
Nodes

.2

OP_Net_Msg

DB_Net_Msg

PM_Net_Msg

FL_Net_Msg

Net_App_Msg

FL_Mgr_Net_Supp_Msg

PM_Mgr_Net_Supp_Msg

Sysexe_Net_Supp_Msg

2.2.2.2
Network Nodes

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_1_Msg
DB_Net_1_Msg
PM_Net_1_Msg
FL_Net_1_Msg
Net_App_Msg
Netexec_Comd_1_Msg
Netnode_1_Status_Msg
SW TASK 171
CPU1 NETNODE
.1

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_5_Msg
DB_Net_5_Msg
PM_Net_5_Msg
FL_Net_5_Msg
Net_App_Msg
Netexec_Comd_5_Msg
Netnode_5_Status_Msg
SW TASK 175
CPU5 NETNODE
.5

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_6_Msg
DB_Net_6_Msg
PM_Net_6_Msg
FL_Net_6_Msg
Net_App_Msg
Netexec_Comd_6_Msg
Netnode_6_Status_Msg
SW TASK 176
CPU6 NETNODE
.6

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_7_Msg
DB_Net_7_Msg
PM_Net_7_Msg
FL_Net_7_Msg
Net_App_Msg
Netexec_Comd_7_Msg
Netnode_7_Status_Msg
SW TASK 177
CPU7 NETNODE
.7

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_2_Msg
DB_Net_2_Msg
PM_Net_2_Msg
FL_Net_2_Msg
Net_App_Msg
Netexec_Comd_2_Msg
Netnode_2_Status_Msg
SW TASK 172
CPU2 NETNODE
.2

ALL CPU NETNODES
ARE INTERCONNECTED

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_8_Msg
DB_Net_8_Msg
PM_Net_8_Msg
FL_Net_8_Msg
Net_App_Msg
Netexec_Comd_8_Msg
Netnode_8_Status_Msg
SW TASK 178
CPU8 NETNODE
.8

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_3_Msg
DB_Net_3_Msg
PM_Net_3_Msg
FL_Net_3_Msg
Net_App_Msg
Netexec_Comd_3_Msg
Netnode_3_Status_Msg
SW TASK 173
CPU3 NETNODE
.3

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_9_Msg
DB_Net_9_Msg
PM_Net_9_Msg
FL_Net_9_Msg
Net_App_Msg
Netexec_Comd_9_Msg
Netnode_9_Status_Msg
SW TASK 179
DC1 NETNODE
.9

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_4_Msg
DB_Net_4_Msg
PM_Net_4_Msg
FL_Net_4_Msg
Net_App_Msg
Netexec_Comd_4_Msg
Netnode_4_Status_Msg
SW TASK 174
CPU4 NETNODE
.4

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_C_Msg
DB_Net_C_Msg
PM_Net_C_Msg
FL_Net_C_Msg
Net_App_Msg
Netexec_Comd_C_Msg
Netnode_C_Status_Msg
SW TASK 17C
DC4 NETNODE
.12

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_B_Msg
DB_Net_B_Msg
PM_Net_B_Msg
FL_Net_B_Msg
Net_App_Msg
Netexec_Comd_B_Msg
Netnode_B_Status_Msg
SW TASK 17B
DC3 NETNODE
.11

PM_Mgr_Net_Supp_Msg
FL_Mgr_Net_Supp_Msg
Sysexe_Net_Supp_Msg
OP_Net_A_Msg
DB_Net_A_Msg
PM_Net_A_Msg
FL_Net_A_Msg
Net_App_Msg
Netexec_Comd_A_Msg
Netnode_A_Status_Msg
SW TASK 17A
DC2 NETNODE
.10

2.3;1
CSCI 12 Data Base

2.3.1;2
Data Base User

SW TASK 181 / CPU1 DB USER (.1)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_1_Msg
- DB_Net_1_Msg

SW TASK 182 / CPU2 DB USER (.2)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_2_Msg
- DB_Net_2_Msg

SW TASK 183 / CPU3 DB USER (.3)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_3_Msg
- DB_Net_3_Msg

SW TASK 184 / CPU1 DB USER (.4)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_4_Msg
- DB_Net_4_Msg

SW TASK 185 / CPU5 DB USER (.5)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_5_Msg
- DB_Net_5_Msg

SW TASK 186 / CPU6 DB USER (.6)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_6_Msg
- DB_Net_6_Msg

SW TASK 187 / CPU7 DB USER (.7)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_7_Msg
- DB_Net_7_Msg

SW TASK 188 / CPU8 DB USER (.8)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_8_Msg
- DB_Net_8_Msg

SW TASK 189 / DC1 DB USER (.9)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_9_Msg
- DB_Net_9_Msg

SW TASK 18A / DC2 DB USER (.10)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_A_Msg
- DB_Net_A_Msg

SW TASK 18B / DC3 DB USER (.11)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_B_Msg
- DB_Net_B_Msg

SW TASK 18C / DC4 DB USER (.12)
- PM_Mgr_DB_Supp_Msg
- FL_Mgr_DB_Supp_Msg
- Sysexe_DB_Supp_Msg
- D-Base_App_Msg
- DB_Mgr_User_C_Msg
- DB_Net_C_Msg

2.3.2;1
Data Base Manager

Sysexe_DB_Supp_Msg

FL_Mgr_DB_Supp_Msg

PM_Mgr_DB_Supp_Msg

DB_Mgr_User_Msg

SW Task 18D
DB Manager 1

.1

DB_Mgr_User_Msg

DB_Mgr_Mgr_Msg

Sysexe_DB_Supp_Msg

FL_Mgr_DB_Supp_Msg

PM_Mgr_DB_Supp_Msg

DB_Mgr_User_Msg

SW Task 18E
DB Manager 2

.2

DB_Mgr_User_Msg

2.4;1
CSCI 13 Local PM

2.5;1
CSCI 14 Local FL

FL_Net_1_Msg
OP_FL_1_Msg

FL_Net_2_Msg
OP_FL_2_Msg

FL_Net_3_Msg
OP_FL_3_Msg

FL_Net_4_Msg
OP_FL_4_Msg

SW TASK 191
CPU1 LOCAL FL
.2

SW TASK 192
CPU2 LOCAL FL
.3

SW TASK 193
CPU3 LOCAL FL
.4

SW TASK 194
CPU4 LOCAL FL
.5

Manager_Local_
FL_1_Msg

Manager_Local_
FL_2_Msg

Manager_Local_
FL_4_Msg

Manager_Local_
FL_3_Msg

SW TASK 195
CPU5 LOCAL FL
.6

FL_Net_5_Msg
OP_FL_5_Msg

Manager_Local_
FL_5_Msg

Manager_Local_
FL_6_Msg

SW TASK 196
CPU6 LOCAL FL
.7

FL_Net_6_Msg
OP_FL_6_Msg

Sysexe_FL_Mgr_Supp_Msg
FL_Mgr_Net_Supp_Msg
FL_Mgr_DB_Supp_Msg

SW TASK 19D
FL MANAGER
.1

Manager_Local_
FL_7_Msg

SW TASK 197
CPU7 LOCAL FL
.8

FL_Net_7_Msg
OP_FL_7_Msg

Manager_Local_
FL_8_Msg

SW TASK 198
CPU8 LOCAL FL
.9

FL_Net_8_Msg
OP_FL_8_Msg

Manager_Local_
FL_9_Msg

Manager_Local_
FL_A_Msg

Manager_Local_
FL_B_Msg

Manager_Local_
FL_C_Msg

SW TASK 199
DC1 LOCAL FL
.10

SW TASK 19A
DC2 LOCAL FL
.11

SW TASK 19B
DC3 LOCAL FL
.12

SW TASK 19C
DC4 LOCAL FL
.13

FL_Net_9_Msg
OP_FL_9_Msg

FL_Net_A_Msg
OP_FL_A_Msg

FL_Net_B_Msg
OP_FL_B_Msg

FL_Net_C_Msg
OP_FL_C_Msg

● **Software Task Description**

        SW Task 101 Det. Beam 1-20
        SW Task 102 Det. Beam 21-40
        SW Task 103 Det. Beam 41-60
        SW Task 104 Det. Beam 62-80
        SW Task 105 Det. Beam 81-100
        SW Task 106 Det. Beam 101-120
        SW Task 107 Det. Management
        SW Task 111 Trk. Beam 1-20
        SW Task 112 Trk. Beam 21-40
        SW Task 113 Trk. Beam 41-60
        SW Task 114 Trk. Beam 61-80
        SW Task 115 Trk. Beam 81-100
        SW Task 121 Class Ch. 1-4
        SW Task 122 Class Ch. 5-8
        SW Task 123 Class Ch. 9-12
        SW Task 124 Class Management
        SW Task 131 Operator 1 Audio
        SW Task 132 Operator 2 Audio
        SW Task 133 Operator 3 Audio
        SW Task 134 Operator 4 Audio
        SW Task 141 Stabilization
        SW Task 151 Time Synch
        SW Task 161 Detection C&D
        SW Task 162 Track C&D
        SW Task 163 Class C&D
        SW Task 164 Backup C&D
        SW Task 301 Sig. Cond. Hyd. 1-400
        SW Task 302 Sig. Cond. Hyd. 401-800
        SW Task 303 Sig. Cond. Hyd. 801-1200
        SW Task 304 Sig. Cond. Hyd. 1201-1600
        SW Task 401 Form Fixed Detect Beams
        SW Task 402 Form Steerable Beams

   ** All Support Function Software will be specified in the future

NAME/TITLE: **SW Task 101 / Detection Beams 1 to 20**

SUBTASK
        None

INPUT MESSAGES
        Detect_Beams_Msg_Beam_1_to_20
        Detect_Selects_Msg
        Time_Msg
        Position_Msg
        Support_Application_Msg

OUTPUT MESSAGES
        Detect_Data_Msg_Beam_1_to_20
        Gain_Control_Signal_Msg_Beam_1_to_20

TASK DESCRIPTION
        - SW Task 101 compares the signal amplitude of each beam (1 to 20) to a series of thresholds
        - Signal amplitude is converted to the corresponding threshold level (i.e. is quantized)
        - The requantized data is integrated over time and formatted for display

DESIGN FACTORS
        Performance
                Processing Required: 0.22 MIPS
                Memory Required: 60 KBytes
                Priority: 3
                Response Time: 250 msec

NAME/TITLE: **SW Task 102 / Detection Beams 21 to 40**
        Same as SW Task 101 except it will process Detection Beams 21 to 40

NAME/TITLE: **SW Task 103 / Detection Beams 41 to 60**
        Same as SW Task 101 except it will process Detection Beams 41 to 60

NAME/TITLE: **SW Task 104 / Detection Beams 61 to 80**
        Same as SW Task 101 except it will process Detection Beams 61 to 80

NAME/TITLE: **SW Task 105 / Detection Beams 81 to 100**
        Same as SW Task 101 except it will process Detection Beams 81 to 100

NAME/TITLE: **SW Task 106 / Detection Beams 101 to 120**
        Same as SW Task 101 except it will process Detection Beams 101 to 120

NAME/TITLE: **SW Task 107 / Detection Management**

SUBTASK
    None

INPUT MESSAGES
    Gain_Control_Signal_Msg_Beam_1_to_20
    Gain_Control_Signal_Msg_Beam_21_to_40
    Gain_Control_Signal_Msg_Beam_41_to_60
    Gain_Control_Signal_Msg_Beam_61_to_80
    Gain_Control_Signal_Msg_Beam_81_to_100
    Gain_Control_Signal_Msg_Beam_101_to_120

OUTPUT MESSAGES
    Gain_Control_Signal_Msg

TASK DESCRIPTION
    SW Task 107 manage the Gain_Control_Signal_Msg of all 120 beams to provide the correct Gain control Signal

DESIGN FACTORS
    Performance
        Processing Required: 0.22 MIPS
        Memory Required: 60 KBytes
        Priority: 3
        Response Time: 250 msec

NAME/TITLE: **SW Task 111 / Track Beams 1 to 20**

SUBTASK
    None

INPUT MESSAGES
    Track_Beams_Msg_Beam_1_to_20
    Track_Selects_Msg
    Time_Msg
    Position_Msg
    Support_Application_Msg

OUTPUT MESSAGES
    Track_Data_Msg_Beam_1_to_20
    CIC_Track_Msg_Beam_1_to_20
    Steering_Data_Msg_Beam_1_to_20
    Support_Application_Msg

TASK DESCRIPTION
    This task will process Track_Beams_Msg_Beam_1_to_20 to:

- Determine the estimate bearing for each assigner tracker
- Calculate beam steering coefficients to recenter each Track Beam on the estimated target position
- Compute the bearing rate for each tracker by a least squares fit of the bearing data over the most recent 30 sec interval

Trackers are initiated on bearings identified by the Track_Selects_Msg parameters

DESIGN FACTORS
 Performance
  Processing Required: 0.22 MIPS
  Memory Required: 60 KBytes
  Priority: 4
  Response Time: 1000 msec

NAME/TITLE: **SW Task 112 / Track Beams 21 to 40**
 Same as SW Task 111 except it will process Track Beams 21 to 40

NAME/TITLE: **SW Task 113 / Track Beams 41 to 60**
 Same as SW Task 111 except it will process Track Beams 41 to 60

NAME/TITLE: **SW Task 112 / Track Beams 61 to 80**
 Same as SW Task 111 except it will process Track Beams 61 to 80

NAME/TITLE: **SW Task 112 / Track Beams 81 to 100**
 Same as SW Task 111 except it will process Track Beams 81 to 100

NAME/TITLE: **SW Task 121 / Class Channel 1 to 4**

SUBTASK
 None

INPUT MESSAGES
 Time_Msg
 Position_Msg
 Class_Beams_Msg_Ch_1_to_4
 Class_Selects_Msg_Ch_1_to_4
 Analysis_Control_Msg

OUTPUT MESSAGES
 Class_Data_Msg_Ch_1_to_4
 Class_Beams_Request_Msg_Ch_1_to_4

TASK DESCRIPTION
 This task will process Class Beam data on channels 1 to 4 to determine the characteristics of the acoustic energy in the selected beam and then those characteristics will be compared to known target signatures
 Analysis is initiated on track beams identified from two sources:
  - by operator via Class_Select_Msg
  - by an automatically selecting process

DESIGN FACTORS
 Performance
  Processing Required: 0.44 MIPS

Memory Required: 85 KBytes
Priority: 6
Response Time: 1000 msec

NAME/TITLE: **SW Task 122 / Class Channel 5 to 8**
          Same as SW Task 121 except it will process information on channel 5 to 8

NAME/TITLE: **SW Task 123 / Class Channel 9 to 12**
          Same as SW Task 121 except it will process information on channel 5 to 8

NAME/TITLE: **SW Task 124 / Class Management**

SUBTASK
          None

INPUT MESSAGES
          Class_Data_Msg_Ch_1_to_4
          Class_Data_Msg_Ch_5_to_8
          Class_Data_Msg_Ch_9_to_12
          Class_Beams_Request_Msg_Ch_1_to_4
          Class_Beams_Request_Msg_Ch_5_to_8
          Class_Beams_Request_Msg_Ch_9_to_12

OUTPUT MESSAGES
          Class_Data_Msg
          Class_Beams_Request_Msg

TASK DESCRIPTION
          This task maintain Class Data and Class Beams Request from different channel to provide the correct information

DESIGN FACTORS
          Performance
                    Processing Required: 0.44 MIPS
                    Memory Required: 85 KBytes
                    Priority: 6
                    Response Time: 1000 msec

NAME/TITLE: **SW Task 131 / Operator # 1 Audio**

SUBTASK
          None

INPUT MESSAGES
          Time_Msg
          Position_Msg
          Audio_Beams_Msg_Op_1
          Audio_Selects_Msg_Op_1
          Support_Application_Msg

OUTPUT MESSAGES
          Audio_Beams_Request_Msg_Op_1
          Sound_Msg_Op_1

Support_Application_Msg

TASK DESCRIPTION
- Operator selected beams (detection beams) are converted to analog signals by reconstruction and filtering of the digital beam data.
- Detection beams to be converted to analog audio signals are identified by the Audio cursor parameters.

DESIGN FACTORS
Performance
Processing Required: 0.11 MIPS
Memory Required: 35 KBytes
Priority: 6
Response Time: 100 msec

NAME/TITLE: **SW Task 132 / Operator # 2 Audio**
Same as SW Task 131 except it will generate selected track beams

NAME/TITLE: **SW Task 133 / Operator # 3 Audio**
Same as SW Task 131 except it will generate selected class beams

NAME/TITLE: **SW Task 134 / Operator # 4 Audio**
Operator 4 is a backup position, at any given time, he can perform the same service as any of the other three operator does.

NAME/TITLE: **SW Task 141 / Stabilization**

SUBTASK
None

INPUT MESSAGES
FIX_Msg
Support_Application_Msg

OUTPUT MESSAGES
Position_Msg
Support_Application_Msg

TASK DESCRIPTION
Stabilization creates a position message which includes the instantaneous position and attitude vectors describing ship and sensor array latitude, longitude, heading/heading rate, roll/roll rate, pitch/pitch rate, speed, and depth/depth rate. These parameters are calculated based upon inputs from the ship's navigation system.

DESIGN FACTORS
Performance
Processing Required: 0.12 MIPS
Memory Required: 22 KBytes
Priority: 1
Response Time: 250 msec

NAME/TITLE: **SW Task 151 / Time Synch**

SUBTASK
None

INPUT MESSAGES
>   GMT_Msg
>   Support_Application_Msg

OUTPUT MESSAGES
>   Sample_Synch_Msg
>   Time_Msg
>   Support_Application_Msg

TASK DESCRIPTION
>   Time_Synchronization provides the current time through out the system and generates the synchronization pulses which coordinate the signal conditioner and the beamformer. These parameters are calculated based upon inputs from the ship's navigation system.

DESIGN FACTORS
>   Performance
>>   Processing Required: 0.09 MIPS
>>   Memory Required: 22 KBytes
>>   Priority: 1
>>   Response Time: 2 msec

NAME/TITLE: **SW Task 161 / Detection C&D**

SUBTASK
>   None

INPUT MESSAGES
>   Detect_Data_Msg
>   Sound_Msg_Op_1
>   Sound_Msg_Op_4
>   Support_Application_Msg

OUTPUT MESSAGES
>   Detect_Selects_Msg
>   Audio_Selects_Msg_Op_1
>   Audio_Selects_Msg_Op_4
>   Analysis_Control_Msg
>   Support_Application_Msg

TASK DESCRIPTION
>   This task provides the mechanism for detection information generated by the system to be formatted for evaluation and analyzed (either automatically or by an operator). This function also provides a mechanism whereby options selected based upon this analysis are communicated to the signal processing function.

DESIGN FACTORS
>   Performance
>>   Processing Required: 0.42 MIPS
>>   Memory Required: 150 KBytes
>>   Priority: 6
>>   Response Time: 100 msec

NAME/TITLE: **SW Task 162 / Track C&D**
>   Same as SW Task 161 except it will generate Track_Data information

NAME/TITLE: **SW Task 163 / Class C&D**
>   Same as SW Task 161 except it will generate Class_Data information

NAME/TITLE: **SW Task 164 / Backup C&D**
          Can be any of the other three C&D task at a given time

NAME/TITLE: **SW Task 301 / Sig. Cond. Hyd. 1-400**

SUBTASK
     None

INPUT MESSAGES
     Gain_Control_Signal_Signal_Msg
     Sample_Synch_Msg
     Electronic_Signal_Msg_Hyd_Ch_1_to_400
     Support_Application_Msg

OUTPUT MESSAGES
     Sample_Msg_Hyd_Ch_1_to_400
     Support_Application_Msg

TASK DESCRIPTION
     TBS

DESIGN FACTORS
     TBS

NAME/TITLE: **SW Task 302 / Sig. Cond. Hyd. 401-800**
          Same as SW Task 301 except Hydrophone 401-800
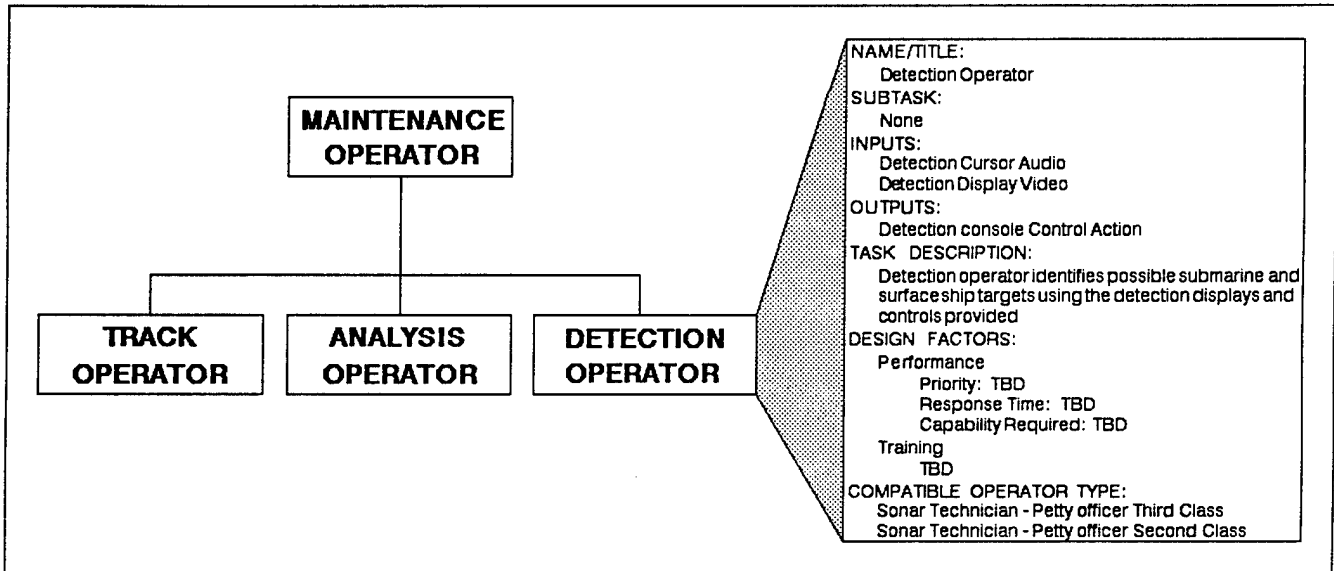
NAME/TITLE: **SW Task 303 / Sig. Cond. Hyd. 801-1200**
          Same as SW Task 301 except Hydrophone 801-1200

NAME/TITLE: **SW Task 304 / Sig. Cond. Hyd. 1201-1600**
          Same as SW Task 301 except Hydrophone 1201-1600

NAME/TITLE: **SW Task 401 / Form Fixed Detect Beams**
     TBS

NAME/TITLE: **SW Task 402 / Form Steerable Beams**
     TBS

Candidate Humanware Architecture


```
MAINTENANCE
OPERATOR


TRACK          ANALYSIS       DETECTION
OPERATOR       OPERATOR       OPERATOR
```

NAME/TITLE:
    Detection Operator
SUBTASK:
    None
INPUTS:
    Detection Cursor Audio
    Detection Display Video
OUTPUTS:
    Detection console Control Action
TASK DESCRIPTION:
    Detection operator identifies possible submarine and
    surface ship targets using the detection displays and
    controls provided
DESIGN FACTORS:
    Performance
        Priority: TBD
        Response Time: TBD
        Capability Required: TBD
    Training
        TBD
COMPATIBLE OPERATOR TYPE:
    Sonar Technician - Petty officer Third Class
    Sonar Technician - Petty officer Second Class

NAME/TITLE
Detection Operator

SUBTASK
None

INPUTS
Detection Cursor audio and detection display video
OUTPUTS

Detection console control actions

TASK DESCRIPTION
Detection operator identifies possible submarine and surface ship targets using the detection displays and controls provided

DESIGN FACTORS
Performance
Priority: TBD
Response Time: TBD
Capability Required:TBD
Training
TBD

COMPATIBLE OPERATOR TYPES

Sonar Technician - Petty Officer Third Class
Sonar Technician - Petty Officer Second Class

NAME/TITLE
Track Operator

SUBTASK
None

INPUTS
Track cursor audio and track display video
OUTPUTS

Track console control actions

TASK DESCRIPTION
Track operator assigns and monitors tracks on possible submarine and surface ship targets (identified by the detection operator) using the track displays and controls provided.

DESIGN FACTORS
Performance
Priority: TBD
Response Time: TBD
Capability Required:TBD
Training
TBD

COMPATIBLE OPERATOR TYPES

Sonar Technician - Petty Officer Third Class
Sonar Technician - Petty Officer Second Class

NAME/TITLE
Classification Operator

SUBTASK
None

INPUTS
Classification cursor audio and classification display video
OUTPUTS

Classification console control actions

TASK DESCRIPTION
Classification operator evaluate and classifies possible submarine and surface ship targets (identified by the detection operator and track by the track operator) using the classification console displays and controls provided.

DESIGN FACTORS
Performance
Priority: TBD
Response Time: TBD
Capability Required:TBD
Training
TBD

COMPATIBLE OPERATOR TYPES

Sonar Technician - Petty Officer Third Class
Sonar Technician - Petty Officer Second Class

NAME/TITLE
Maintenance Operator

SUBTASK
None

INPUTS
Maintenance cursor audio and maintenance display video
OUTPUTS

Maintenance console control actions

TASK DESCRIPTION
Maintenance operator identifies and evaluates the performance of the system and localizes any identified faults for correction using the maintenance displays and controls provided.

DESIGN FACTORS
Performance
Priority: TBD
Response Time: TBD
Capability Required:TBD
Training
TBD

COMPATIBLE OPERATOR TYPES

Sonar Technician - Petty Officer Third Class
Sonar Technician - Petty Officer Second Class

SECTION 5

ENVIRONMENTAL CAPTURE VIEW

1.0 INTRODUCTION

The Environmental Capture View is defined as the documentation of the concept of operations, scenarios, environmental conditions, external interfaces and other external factors which impact the system under design.

The concept of operations describes the proposed approach for operation of the sample sonar system from the operators perspective. Scenarios describe situations and sequences of external events which the system must address. The environmental conditions under which the system must operate include, geographic, meteorologic, electromagnetic, and acoustic environmental factors. The external interfaces to the system are captured including existing and planned electronic and human interfaces, as well as non-cooperative external systems such as target submarines and surface ships. Other external factors which influence and affect the system under design are also documented including design constraints and guidance imposed by the program development sponsor.

The following paragraphs represent a preliminary draft of the Environmental Capture View for an unclassified passive sonar system example. The sample problem is incomplete at this point in its development, but serves as a vehicle for further definition and refinement of the Environmental Capture View.

2.0 ENVIRONMENTAL CAPTURE VIEW OF THE PASSIVE SONAR SYSTEM

2.1 Concept of Operations

The sample passive sonar system will be used for determining the presence, location, or nature of objects in the sea from underwater sound the objects emit. Passive sonar bases its target detection and estimation on sounds that emanate from the target itself.

The received acoustic waveform from each hydrophone consists of one or more signals and background noise. The hydrophone converts the acoustic waveform to an electronic signal. The signals are amplified, filtered, sampled, and digitized in a signal conditioner. The digitized hydrophone outputs from the signal conditioner are combined by a digital beamformer to form two sets of beams: (1) one set of fixed beams for detection which cover 360 degrees in azimuth and 60 degrees in depression/elevation, and (2) one set of tracking beams which can be independently steered to follow selected targets. Detection and track beam data are then processed to obtain detection and position estimation statistics which are displayed to the operator. Based on the values of these statistics, the detection operator (with the help of the system auto-detect function) decides where targets are located. Detected targets are tracked by a track operator who initiates and monitors the track beams to follow selected targets. The system can also automatically assign trackers to contacts detected automatically providing a semi-automatic detection and tracking capability.

Detected targets are analyzed by a classification operator. Analysis will include distinguishing a signal from a target with regard to the type of target that produced the signal. A target is classified by the signal frequency spectrum and dynamics (e.g., a school of fish versus a submarine).

The passive sonar system is designed to detect and track submarine targets that can operate at any speed from 0 to 15 knots at any depth from 0 to 500 feet. The target sound source is assumed to consist of both low frequency broadband noise and vibrational harmonics from rotating machinery with a nominal rotation rate of 2,400 rpms. Therefore, it is expected that the signal from a submarine contains both broadband a fundamental 40-hertz component and the first three harmonics.

The sample sonar system is to be installed on Naval combatants surface ships including frigates, destroyers and cruisers which have an ASW mission. The system is operated and maintained by active duty Naval personnel. The system is manned full time when underway in a variety of watch conditions. The system provides high performance color displays and synthesized audio to the operators of the processed sonar data and also provides automatic aids for target detection, tracking and classification. Operators conduct a continuous 360 degree search for acoustic contacts and concurrently perform tracking and classification on contacts which are identified as high interest. The system can be operated by two to four persons depending upon the number of acoustic targets and the operational tempo.

When the acoustic contact load is light (e.g., three or fewer contacts at a time) and the operations tempo is low (e.g., independent transit during peace time) two operators can adequately man the system. One operator will typically be dedicated to passive acoustic search using the detection workstation configuration and one operator will divide his time between the tracking and classification tasks.

When the acoustic contact load is moderate (e.g., four to ten contacts at a time) or the operations tempo becomes stressing (e.g., operating within a CVBG during peacetime) three operators are required to man the system. One operator will typically be dedicated to passive acoustic search using the detection workstation configuration, one operator will be dedicated to tracking high interest contacts using the track workstation configuration, and one operator will be dedicated to classifying high interest contacts using the class workstation configuration.

When the acoustic contact load is high (e.g., more than ten contacts at a time) or the operations tempo becomes very stressing (e.g. tracking a treat submarine during a period of rising tensions) four operators are required to man the system. One operator will typically be dedicated to passive acoustic search using the detection workstation configuration, one operator will be dedicated to tracking high interest contacts, and one operator will be dedicated to classifying high interest contacts. A fourth operator will be assigned to support either the primary detection, track or classification operator as required to accommodate the existing workload.

2.1.1 Detection Operator Tasks

TBS

2.1.2 Track Operator Tasks

TBS

2.1.3 Classification Operator Tasks

TBS

2.2 Operational Scenarios

This subsection discusses the expected scenarios to be encountered by the sample sonar system under design both in the near-term and far-term over the system's expected life. The scenarios of most interest are represented by the most likely and most stressing operating situations. These scenarios are provided in enough detail to provide a top-level view of the anticipated events and to gain an appreciation of system interface requirements. This subsection also describes the geographical and environmental factors with which the sample sonar system must contend.

2.2.1 Spectrum of Operational Scenarios

This section attempts to bound and describe the wide spectrum of possible operational scenarios for the sample sonar system. The set of possible scenarios is very large and can not be efficiently represented by a list of scenario cases. The approach we will employ here is to establish the key parameters which characterize the scenarios of interest and then to establish the range of possible cases for each parameter. These parameters provide a mechanism to characterize entire classes of scenarios and to identify the most likely and most stressing cases to support system simulation and analysis.

The following four key parameters describe the spectrum of operational scenarios which the sample passive sonar system is intended to operate:

(1) Tempo of Operations - Defined as the level of operational intensity associated with the platform mission and the system's role in supporting that mission. The range of possible cases for the Tempo of Operations scenario parameter is illustrated as follows:

Low - Independent transit in US waters
Moderate - CVBF screening operations in open ocean
High - Threat submarine track/trail operations

(2) Level of Conflict - Defined as the combat readiness or alert state of the platform and related to the likelihood of hostile actions. The range of possible cases for the Level of Conflict scenario parameter is illustrated as follows:

Peacetime
Crisis Response
Transition to War
Regional Conflict
Global Conventional War

(3) Environmental Acoustic Conditions - Defined as the description of prevailing environmental conditions which affect acoustic sensor performance. The range of possible cases for the Environmental Acoustic Conditions scenario parameter is illustrated as follows:

Shallow vs deep water
High vs low ambient noise
High vs low propagation loss (DP/CZ/BB/ducted environments)

(4) Acoustic Contact Density - Defined as the number of contacts within acoustic detection range of the system. The range of possible cases for the Acoustic Contact Density scenario parameter is illustrated as follows:

Low - less than 3 simultaneous acoustic contacts
Moderate - 3 to 10 simultaneous acoustic contacts
High - 10 to 15 simultaneous acoustic contacts
Very High - greater than 15 simultaneous acoustic contacts

2.2.2 Most Likely Scenarios

The set of most likely (or most common) scenarios which the sample passive sonar system will encounter is derived by examining the spectrum of possible scenarios. The most likely range for each of the scenario factors is determined and then the most likely combinations of the four parameters are identified. Understanding the set of most likely scenarios represents an important aspect in developing the system design. These scenarios also define one set of test cases which can be used to simulate and analyze the system under design.

The following three scenarios represent the most likely scenarios for the sample sonar system:

TBS

2.2.3 Most Stressing Scenarios

The set of most stressing scenarios which the sample passive sonar system will encounter is also derived by examining the spectrum of possible scenarios. The most stressing cases for each of the scenario factors is determined and then the most stressing combinations of the four parameters are identified. Understanding the set of most stressing scenarios represents an important aspect in developing a system design which is operable and meets the intent of the system requirements. These scenarios also define a second set of test cases which can be used to simulate and analyze the system under design.

The following three scenarios represent the most stressing scenarios for the sample sonar system:

TBS

2.3 External Systems and Interfaces

This subsection summarizes the system boundaries, external interfaces to the system, and describes how the sample passive sonar system fits into the overall platform architecture and organization. It does not address the internal structure of the system, but serves to define the system's boundaries and relationships to other systems and activities which are considered outside the scope of the system under design. It describes and defines the objects external to the system and the interfaces of those objects to the system under design.

2.3.1 Friendly Systems and Cooperative Interfaces

2.3.1.1 Navigation System

2.3.1.1.1 Navigation System Outputs (to sample sonar system)

The sample sonar system receives two navigation system inputs via a serial RS-232 interface: time and position.

NAME: **Position**

STRUCTURE/DESIGN FACTOR:

Component	Type	Unit	Range	Increment	Size	Periodicity	Accuracy	Format
Longitude	data	deg min sec	0 – 360	0.01 sec	9 Bytes	62.5 msec	0.01 sec	DDD MM SS.SS
Latitude	data	deg min sec	0 – 90 N/S	0.01 sec	9 Bytes	62.5 msec	0.01 sec	DDD MM SS.SS
Heading	data	deg	0 – 360	0.1 deg	4 Bytes	62.5 msec	0.1 deg	DDD.D
Heading Rate	data	deg/sec	0 – +/-10	0.1 deg/sec	4 Bytes	62.5 msec	0.1 deg/sec	s DD.D
Roll	data	deg (P/S)	0 – 80	0.1	4 Bytes	62.5 msec	0.1	DD.D O
Roll Rate	data	deg/sec (P/S)	0 – 10	0.1	4 Bytes	62.5 msec	0.1	DD.D O
Pitch	data	deg (U/D)	0 – 80	0.1	4 Bytes	62.5 msec	0.1	DD.D O
Pitch Rate	data	deg/sec (P/S)	0 – 10	0.1	4 Bytes	62.5 msec	0.1	DD.D O
Velocity	data	mile/hr	0 – +/-50	0.1	4 Bytes	62.5 msec	0.1	s VV.V
Acceleration	data	(mile/hr)/hr	0 – +/-100	0.1	5 Bytes	62.5 msec	0.1	s AAA.A
Depth	data	feet	0 – 2000	0.1	6 Bytes	62.5 msec	0.1	dddd.d
Depth Rate	data	feet/sec	0 – +/- 100	0.1	5 Bytes	62.5 msec	0.1	s ddd.d

NAME: **Time**

STRUCTURE/DESIGN FACTOR:

Component	Type	Unit	Range	Increment	Size	Periodicity	Accuracy	Format
Day	data	day	0 – 31	1	2 Bytes	250 msec	1	DD
Hour	data	hr	0 – 24	1	2 Bytes	250 msec	1	HH
Minute	data	min	0 – 60	1	2 Bytes	250 msec	1	MM
Second	data	sec	0 – 60	0.01	4 Bytes	250 msec	0.01	SS.SS

2.3.1.1.2 Navigation System Inputs (from sample sonar system)
TBS

2.3.1.1.3 Navigation System Functional Description
TBS

2.3.1.2 Combat Information Center

2.3.1.2.1 Combat Information Center Outputs (to sample sonar system)
TBS

2.3.1.2.2 Combat Information Center Inputs (from sample sonar system)

The sample sonar system sends target tracking data to the ship's Combat Information Center via serial RS-232 interface.

NAME: **CIC_Track**

STRUCTURE/DESIGN FACTOR:

Component	Type	Unit	Range	Increment	Size	Periodicity	Accuracy	Format
Time	data	hr min sec	0 -- 24 hr	1 sec	8 Bytes	250 msec	0.01 sec	HH MM SS.SS
CIC Track Beam 1								
Track ID	data	integer	0 -- 100	1	3 Bytes	1 sec	N/A	NNN
°	°	°	°	°	°	°	°	°
°	°	°	°	°	°	°	°	°
CIC Track Beam 100								
Track ID	data	integer	0 -- 100	1	3 Bytes	1 sec	N/A	NNN
CIC Track Beam 1								
Bearing	data	deg	0 -- 360	0.1	4 Bytes	1 sec	0.1	DDD.D
°	°	°	°	°	°	°	°	°
°	°	°	°	°	°	°	°	°
CIC Track Beam 100								
Bearing	data	deg	0 -- 360	0.1	4 Bytes	1 sec	0.1	DDD.D
CIC Track Beam 1								
Signal to Noise Ratio	data	dB	0 -- +/-35	0.1	4 Bytes	1 sec	0.1	s RR.R
°	°	°	°	°	°	°	°	°
°	°	°	°	°	°	°	°	°
CIC Track Beam 100								
Signal to Noise Ratio	data	dB	0 -- +/-35	0.1	4 Bytes	1 sec	0.1	s RR.R

2.3.2 Hostile Systems and Non-cooperative Interfaces

2.3.2.1 Submarine Surface Ship and Torpedo Radiated Noise

The sources of radiated noise on ships, submarines and torpedoes an be grouped into three major classes: machinery noise, propeller noise and hydrodynamic noise. Machinery noise originates as mechanical vibration and resonance of the many mechanical systems onboard a vessel. Propeller noise is distinct in that it originates outside the hull and contains both mechanical and flow relates components. Hydrodynamic noise originates in the irregular and fluctuating flow of water past the hull of the moving vessel.
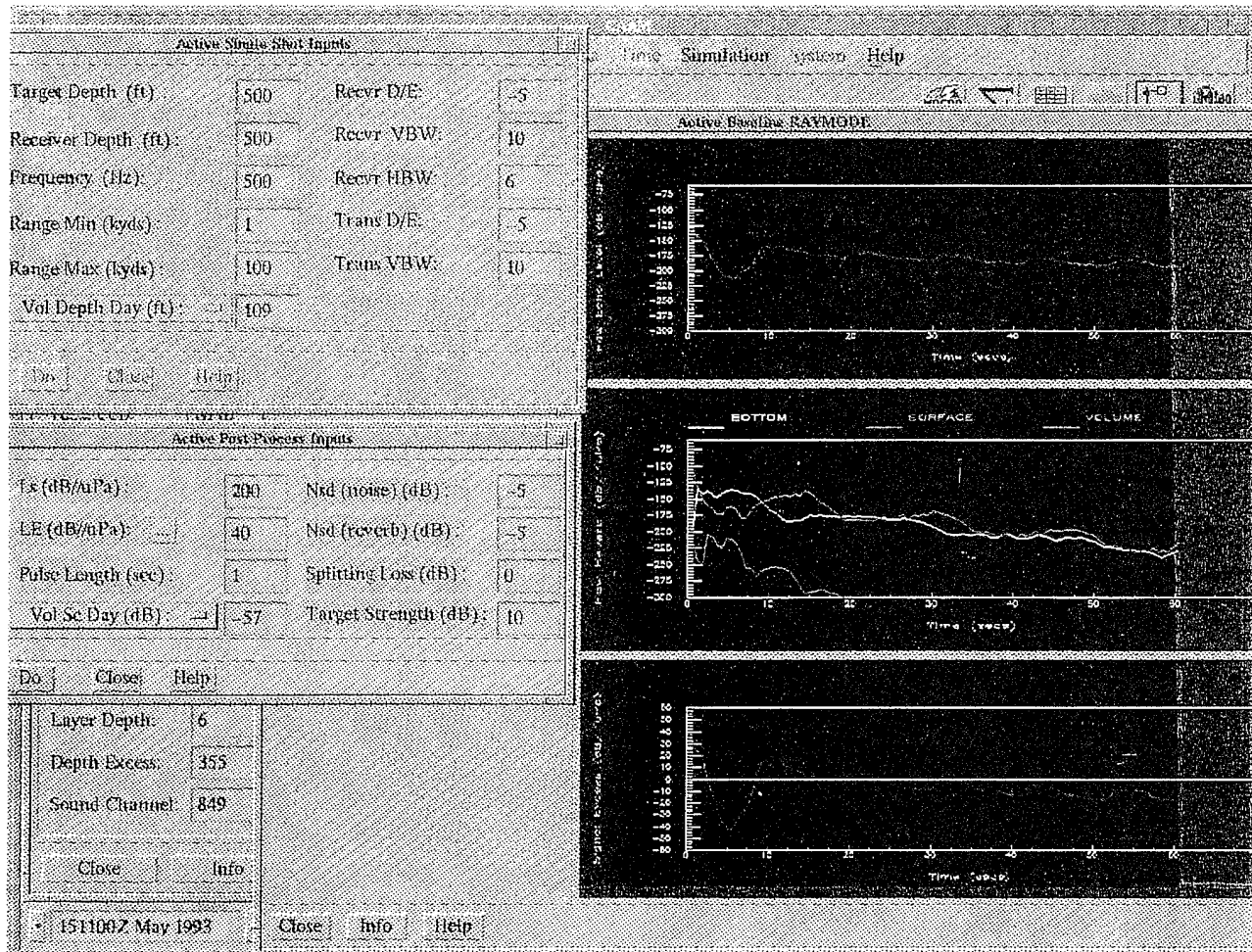
2.4 Environmental Conditions

The sample sonar system operates in a wide variety of acoustic environmental conditions which vary geographically and through the months of the year. This section captures representative acoustic environments in terms of sound velocity profiles, ambient noise, propagation loss, and bottom type. This environmental data is required to model the detection performance of the sample sonar system.

2.4.1 Sound Velocity Profiles

Representative sound velocity profiles, as shown below, are captured in this section for each of the environments listed in section 2.4 above. (Only one representative SVP is actually provided to limit the size of the example problem documentation.
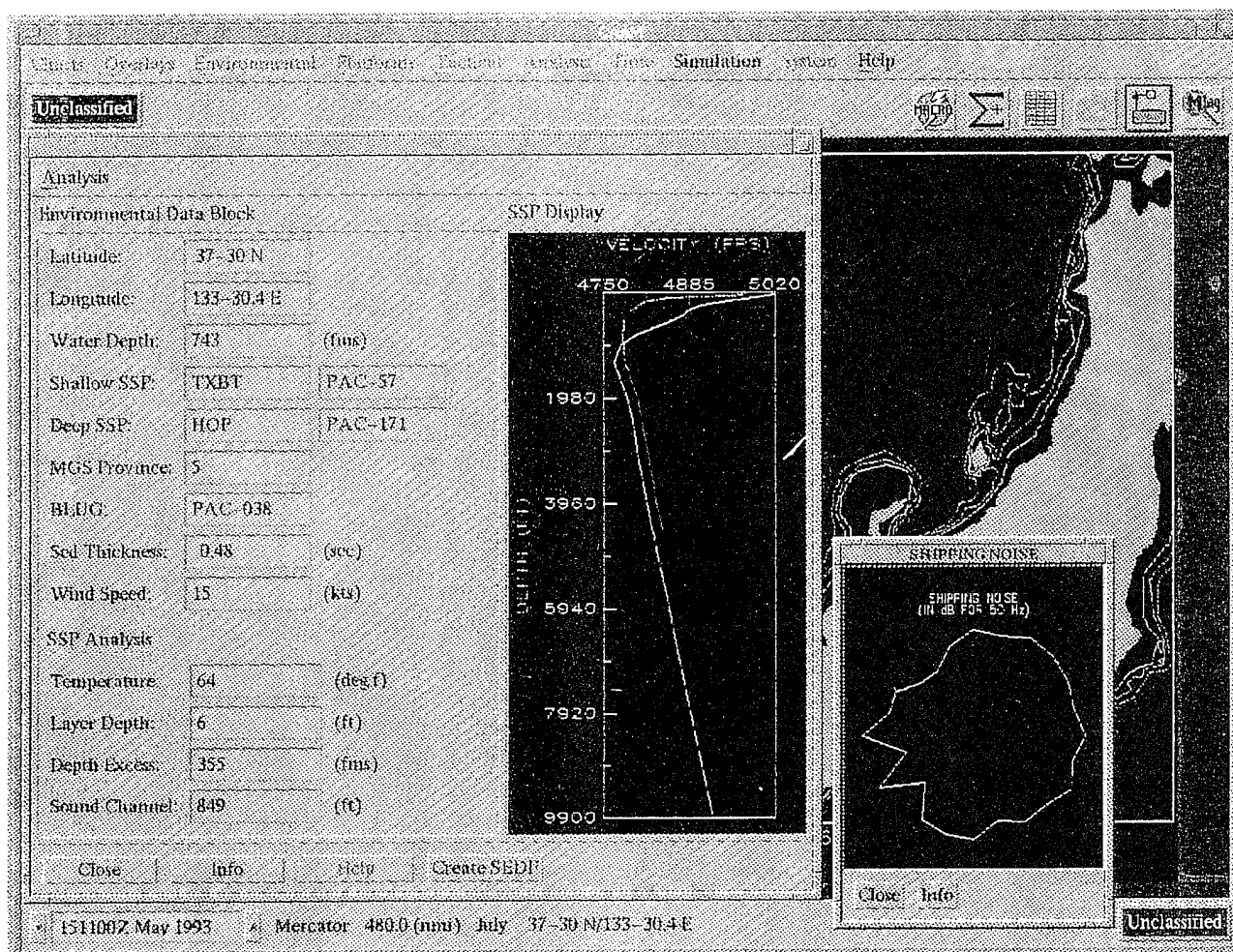
2.4.2 Ambient Noise

Representative ambient noise tables are captured in this section for each of the environments listed in section 2.4 above. (Only one representative ambient noise table is actually provided to limit the size of the example problem documentation.

SOUND VELOCITY DISPLAY

2.4.3 Propagation Loss

Representative propagation loss curves, as shown below, are captured in this section for each of the environments listed in section 2.4 above. (Only one representative propagation loss curve is actually provided to limit the size of the example problem documentation.

PROPAGATION LOSS DISPLAY

2.4.4 Bottom Types

Representative bottom types are captured for each of the environments listed.

Description	Lat/Long (Deg)	Depth (Ft)	Bottom Class	Layer Depth (Ft)	Peak BV Freq (cph)
Central Gulf of Oman	23.3N/62.OE	10,821	2	33	16.4
South of the Strait of Hormu	25.45N/56.5E	443	2	33	14
South Central Persian Gulf	25.3N/53.3E	148	2	0	17
North Central Persian Gulf	28.3N/50.OE	167	2	0	27.5
North Central Arabian Sea	20.0N/62.3E	11,434	5	33	16.2
Central Arabian Sea	15.0N/62.3E	13,094	4	98	18.6
Central Red Sea	22.3N/38.0E	2,627	7	66	10.6

DISTRIBUTION

	Copies		Copies
DOD ACTIVITIES (CONUS)		ATTN GIFT & EXCHANGE DIVISION LIBRARY OF CONGRESS WASHINGTON DC 20540	4
DEFENSE TECH INFORMATION CTR 8725 JOHN J KINGMAN RD SUITE 0944 FT BELVOIR VA 22060-6218	12	TRIDENT SYSTEMS INCORPORATED ATTN NICHOLAS KARANGELEN 10201 LEE HIGHWAY SUITE 300	5
ATTN CODE AS/RA (BALA RAMESH) ADMINISTRATIVE SCIENCES DEPT MONTEREY CA 93943	1	FAIRFAX VA 22030	
ATTN CODE 5543 (KATHERINE MEADOWS) NAVAL RESEARCH LABORATORY 4555 OVERLOOK AVE SW WASHINGTON DC 20375	1	ATTN ADRIEN MESKIN ATR 15210 DINO DRIVE BURTONSVILLE MD 20866-1172	2
ATTN CODE 4411B (ELIZABETH WALD) OFFICE OF NAVAL RESEARCH 800 NORTH QUINCY STREET ARLINGTON VA 22217-5000	1	ATTN LONNIE WELCH THE REAL-TIME COMPUTING LAB DEPT OF COMPUTER AND INFORMATION SCIENCE NJIT UNIVERSITY HEIGHTS NEWARK NJ 07102	1

ATTN LONNIE WELCH ... see above

NON-DOD ACTIVITIES		INTERNAL	
		A	1
		A44 (R SCALZO)	1
		B	1
THE CNA CORPORATION PO BOX 16268 ALEXANDRIA VA 22302-0268	2	B02	1
		B05 (H CRISP)	5
		B10 (S BARKER)	1
		B10 (W FARR)	1
		B10 (W MCCOY)	1

DISTRIBUTION (Continued)

Copies

B20	1
B30	1
B35 (M CHANG)	1
B44	1
B44 (N HOANG)	20
B44 (S HOWELL)	2
B44 (M JENKINS)	1
B44 (C NGUYEN)	1
B44 (H ROTH)	1
B44 (M WILSON)	1
D	1
D4	1
E231	3
E232	2
F	1
G	1
K	1
L	1
N	1
N24 (R HOLDEN)	1
N72D (GIDEP)	1